

501P0605US00

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT



別紙添付の書類に記載されている事項は下記の出願書類に記載されて
いる事項と同一であることを証明する。 #5
7-9-03

This is to certify that the annexed is a true copy of the following application as filed with this Office. JM

出 願 年 月 日
Date of Application:

2000年 4月21日

出 願 番 号
Application Number:

特願2000-120940

出 願 人
Applicant(s):

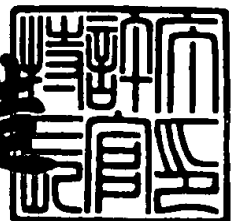
ソニー株式会社

CERTIFIED COPY OF
PRIORITY DOCUMENT

2001年 3月 2日

特 許 庁 長 官
Commissioner,
Patent Office

及 川 耕 造



出証番号 出証特2001-3014245

【書類名】 特許願

【整理番号】 0000221906

【提出日】 平成12年 4月21日

【あて先】 特許庁長官 近藤 隆彦 殿

【国際特許分類】 H04L 12/00
H04L 29/00

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 山岸 靖明

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 権野 善久

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 西尾 郁彦

【発明者】

【住所又は居所】 東京都品川区北品川 6 丁目 7 番 3 5 号 ソニー株式会社
内

【氏名】 角田 智弘

【特許出願人】

【識別番号】 000002185

【氏名又は名称】 ソニー株式会社

【代表者】 出井 伸之

【代理人】

【識別番号】 100082762

【弁理士】

【氏名又は名称】 杉浦 正知

【電話番号】 03-3980-0339

【手数料の表示】

【予納台帳番号】 043812

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9708843

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法

【特許請求の範囲】

【請求項 1】 公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信装置において、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する管理手段と、

上記管理手段によって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、

上記検出手段で検出された上記差分情報を送信する送信手段とを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信装置。

【請求項 2】 請求項 1 に記載の送信装置において、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする送信装置。

【請求項 3】 請求項 1 に記載の送信装置において、

上記コンテナエントリおよび／または上記リーフエントリの属性に、上記最新の公開鍵証明情報および上記最新の公開鍵証明情報を取得するための情報のうち何れか一方を選択して格納可能としたことを特徴とする送信装置。

【請求項 4】 請求項 3 に記載の送信装置において、

上記検出手段により上記差分が検出された更新事象からの時間に応じて、上記属性に格納される情報を、上記最新の公開鍵証明情報と上記最新の公開鍵証明情報を取得するための情報との間で変更可能なようにしたことを特徴とする送信装

置。

【請求項 5】 公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信方法において、

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する管理のステップと、

上記管理のステップによって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、

上記検出のステップで検出された上記差分情報を送信する送信のステップとを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信方法。

【請求項 6】 送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信装置において、

送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められた上記ディレクトリの階層構造の変化の差分からなる差分情報を受信する受信手段と、

上記受信手段で受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する管理手段と、

上記差分情報を選択的に取り込み、上記管理手段で管理される上記ディレクトリの階層構造を変更する変更手段とを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵

証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されて上記送信されることを特徴とする受信装置。

【請求項 7】 請求項 6 に記載の受信装置において、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする受信装置。

【請求項 8】 請求項 6 に記載の受信装置において、

上記変更手段は、上記公開鍵証明情報を取得するための証明情報パスに対応する上記コンテナエントリおよび／または上記リーフエントリの更新情報を上記選択的に取り込むことを特徴とする受信装置。

【請求項 9】 送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信方法において、

送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報に対応付け、上記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報に対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められた上記ディレクトリの階層構造の変化の差分からなる差分情報を受信する受信のステップと、

上記受信のステップで受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する管理のステップと、

上記差分情報を選択的に取り込み、上記管理のステップで管理される上記ディレクトリの階層構造を変更する変更のステップとを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されて上記送信されることを特徴とする受信方法。

【請求項 10】 公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信システムにおいて、

自分の配下の情報を格納可能なコンテナエントリに認証局情報に対応付け、上

記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する第 1 の管理手段と、

上記第 1 の管理手段によって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、

上記検出手段で検出された上記差分情報を送信する送信手段と、

上記送信手段により送信された、上記差分情報を受信する受信手段と、

上記受信手段により受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する第 2 の管理手段と、

上記差分情報を選択的に取り込み、上記第 2 の管理手段で管理される上記ディレクトリの階層構造を変更する変更手段とを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信システム。

【請求項 1 1】 請求項 1 0 に記載の送受信システムにおいて、

上記失効情報は、上記公開鍵証明情報のシリアル番号であることを特徴とする送受信システム。

【請求項 1 2】 請求項 1 0 に記載の送受信システムにおいて、

上記コンテナエントリおよび／または上記リーフエントリの属性に、上記最新公開鍵証明情報および上記最新の公開鍵証明情報を取得するための情報のうち何れか一方を選択して格納可能として送信するようにしたことを特徴とする送受信システム。

【請求項 1 3】 請求項 1 2 に記載の送受信システムにおいて、

上記検出手段により上記差分が検出された更新事象からの時間に応じて、上記属性に格納される情報を、上記最新の公開鍵証明情報と上記最新の公開鍵証明情報を取得するための情報との間で変更可能なようにして送信することを特徴とする送受信システム。

【請求項 1 4】 請求項 1 0 に記載の送受信システムにおいて、

上記変更手段は、上記公開鍵証明情報を取得するための証明情報パスに対応する上記コンテナエントリおよび／または上記リーフエントリの更新情報を上記選択的に取り込むことを特徴とする送受信システム。

【請求項 1 5】 公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信方法において

自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、上記コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、上記コンテナエントリと上記リーフエントリとからなるディレクトリの階層構造を管理する第 1 の管理のステップと、

上記第 1 の管理のステップによって管理される上記ディレクトリの階層構造の変化を検出し、該検出結果に基づき上記ディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、

上記検出のステップで検出された上記差分情報を送信する送信のステップと、

上記送信のステップにより送信された、上記差分情報を受信する受信のステップと、

上記受信のステップにより受信された上記差分情報に基づき構成されるディレクトリの階層構造を管理する第 2 の管理のステップと、

上記差分情報を選択的に取り込み、上記第 2 の管理のステップで管理される上記ディレクトリの階層構造を変更する変更のステップとを有し、

上記コンテナエントリおよび／または上記リーフエントリには、最新の公開鍵証明情報を取得可能な情報と、該最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信方法。

【発明の詳細な説明】

【0 0 0 1】

【発明の属する技術分野】

この発明は、この発明は、階層構造を有すると共に、ネットワーク上に分散されて配置されたデータを一方向的に配信する系において、分散された公開鍵ディレクトリエントリの複製の更新を行うような送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法に関する。

【 0 0 0 2 】

【従来の技術】

データの配信方法としては、種々の方法が提案されており、例えば、現在のインターネット上においては、http (Hyper Text Transfer Protocol) のような、TCP / IP (Transmission Control Protocol / Internet Protocol) を基本とするプロトコルが採用されている。TCP / IP では、データの配信を受ける受信側からデータの送信側に対して発呼が行われ、さらに、データの送受信を行う毎に、送信側と受信側との間でコネクションが確立されるようになっている。そのため、信頼性の高いデータの配信を行うことができる。

【 0 0 0 3 】

その反面で、送信側やネットワークの負荷が大きくなり、効率的なデータ配信を行うことが困難になる場合があった。すなわち、データの提供を受ける端末が増加し、データを提供するサーバへのアクセスが集中すると、サーバやネットワークに多大な負荷がかかり、データを要求しても、そのデータを得るまでに多大な時間を要することがあった。

【 0 0 0 4 】

そこで、データの配信を、例えば、広い地域にわたって一斉同報が可能な衛星回線やCATV (Cable Television) 回線、実用化が予定されている地上波デジタル放送などを用いて行う方法が提案されている。この場合、端末の増加によって、サーバやネットワークに対する負荷が影響を受けることがない。

【 0 0 0 5 】

一方、近年では、インターネットなどデジタル通信ネットワークの発達により、ネットワーク上に膨大なデータが蓄積されるようになり、この膨大なデータを効率的に利用することが求められている。そこで、ネットワーク上に分散されて存在するデータを階層的に管理し、ユーザに提供する、ディレクトリサービス

が注目を集めている。ディレクトリサービスを利用することで、ユーザは、ネットワーク上に分散して存在する情報の中から、自分が必要な情報を素早く見つけ出し、アクセスすることが可能になる。

【0006】

ディレクトリサービスは、例えば、国際標準であるOSI (Open Systems Interconnection)などによって、X. 500シリーズとして規定されている。X. 500によれば、ディレクトリについて、開放型システムの集合体であり、各開放型システムが協調して、現実世界のオブジェクトの集合に関する情報の、論理的なデータベースを持つと定義されている。

【0007】

X. 500によるディレクトリサービスによれば、ディレクトリが擁する情報の検索や閲覧を行うことができる。また、例えば電話帳に例えられる一覧やユーザの認証などもディレクトリサービスで提供される。さらに、ディレクトリサービスでは、特に人間のユーザにとって覚え易く、推測ならびに認識し易いように、オブジェクトの名前が付けられる。

【0008】

なお、このX. 500によるディレクトリサービスは、極めて包括的なものであって、プログラムサイズが非常に大きく、TCP/IPをプロトコルとするインターネットでは実現が非常に難しい。そのため、LDAP (Lightweight Directory Access Protocol) といった、TCP/IP向けに軽量化されたディレクトリサービスが提案されている。

【0009】

ディレクトリサービスをユーザが利用する場合、ディレクトリに対してフィルタ処理を行うことができる。フィルタ処理を行うフィルタリングマスクは、ユーザの、ディレクトリへのアクセスの傾向や嗜好に応じて設定される。例えば、ユーザの嗜好に応じて特定の情報ジャンルに対してフィルタリングマスクが設定される。ユーザは、フィルタリングマスクが設定されたディレクトリ情報を、選択的にアクセスすることができる。フィルタ処理を行うことにより、ユーザが不要な情報を保持する必要が無くなる。

【 0 0 1 0 】

近年では、上述した一斉同報を行うデータ伝送手段、すなわち、衛星回線やC A T V回線、地上波デジタル放送などで、このディレクトリサービスを行うことが提案されている。この場合には、ディレクトリサービスによる情報が一方的に提供され、ユーザ側からの情報の要求ができない。したがって、ディレクトリサービスによるディレクトリ情報は、同一の情報が繰り返し送信される。

【 0 0 1 1 】

ユーザ側では、送信されてきた情報を、例えばテレビジョン受像機などに接続して用いられる、デジタル放送用受信機であるI R D (Integrated Reciever D ecoder) や、S T B (Set Top Box) に蓄積する。このとき、上述したフィルタリングマスクを用いてフィルタ処理がなされ、ユーザの嗜好に応じたディレクトリ情報が選択的に蓄積される。フィルタ処理に用いられるフィルタリングマスクは、ユーザ側で設定される。

【 0 0 1 2 】

ところで、オープンなネットワークであるインターネットは、情報の盗聴、改竄、なりすましといった危険が常に伴う。これを防ぐのが認証システムであり、この認証システムを運用するための基盤となる環境を公開鍵基盤(P K I : Publi c Key Infrastructure)と呼ぶ。P K I は、インターネット上の暗号化通信や電子メール、各種決済プロトコルなどを運用するための基盤環境として注目を集めており、その環境の信頼性が将来のインターネット上の商取引(EC: Electronic Commerce)の成否を左右するものと考えられている。

【 0 0 1 3 】

P K I で用いられる公開鍵暗号化方式は、秘密鍵および公開鍵の2つの鍵を用い、暗号化と復号化をこれら2つの別々の鍵で行う、非対称型の暗号化方式である。自分で保管する秘密鍵から公開鍵が作成され、公開鍵は、相手に渡すことができる。公開鍵で暗号化されたものは対応する秘密鍵でしか復号化できず、秘密鍵で暗号化されたものは対応する公開鍵でしか復号化できない。この特性を利用して、電子署名を行うことができる。

【 0 0 1 4 】

また、公開鍵だけでは上述の「なりすまし」が可能なので、認証局（CA:Certificate Authority）による公開鍵証明書（デジタル証明書）の発行を以て、公開鍵の管理とその認証が行われる。公開鍵証明書の発行は、例えば以下のようになされる。加入者が秘密鍵と公開鍵のペアを所定の方法で作成し、その公開鍵を認証局に提出する。認証局は、加入者の身分などを確認してから、証明書のシリアル番号、加入者の名前、公開鍵、有効期間などの情報に認証局の署名をし、認証局が予め公開している公開鍵を利用して電子署名を行った、公開鍵証明書を発行する。

【0015】

図28を用いて、公開鍵証明書の仕組みについて説明する。図28Aは、公開鍵証明書の作成の手順を概略的に示す。公開鍵証明書は、シリアル番号、発行者（認証局）名、有効期限、所有者名などの情報（ここでは証明書情報と称する）および所有者の公開鍵と、証明書情報をダイジェスト関数で変換して得られた「指紋」を認証局の秘密鍵で暗号化した認証局の署名とからなる。なお、ダイジェスト関数は、長い文書を短い決まった長さに変換する、元の情報が少しでも変われば変換結果が大きく異なる、という特徴を持つ関数である。このことから、ダイジェスト関数で得られた値は、元の文書の「指紋」と称される。

【0016】

図28Bは、このようにして作成された公開鍵証明書を確認する手順を概略的に示す。公開鍵証明書に含まれる証明書情報からダイジェスト関数により「指紋」が求められる。一方、公開鍵証明書において、証明書情報に添付される署名が認証局の公開鍵で復号化される。この復号化された結果と、上述の「指紋」とが比較され、両者が一致すれば、公開鍵証明書が改竄されていない正当なものであると判断することができる。

【0017】

【発明が解決しようとする課題】

現在、PKIの重要な機能と位置付けられる、上述の公開鍵証明書を、広範囲に分散したユーザに対して配布管理するシステムの完備が急がれている。しかしながら、従来では、失効した公開鍵証明書の通知を、広範囲に分散した大多数の

ユーザに効果的に通知する方式が確立されていないという問題点があった。

【 0 0 1 8 】

例えば、証明書を盗難された場合や、個人が退職してその証明書を利用する資格を失った場合など、公開鍵証明書に定められている有効期限を待たずに、その公開鍵証明書を無効化したいことがある。

【 0 0 1 9 】

このような場合、無効化された（失効した）証明書のシリアル番号や失効した日付などを、公開鍵証明書を配布・管理するシステムに登録しておき、認証手続きの際に公開鍵証明書を利用するアプリケーションに対して、対象の公開鍵証明書が失効していないかを毎回確認させるようにしなければならない。この公開鍵証明書の失効情報、さらに公開鍵証明書そのものを配布、管理および失効照会を提供するシステムとして、オンライン失効情報照会（PKIディレクトリサービス）サービスが考えられる。

【 0 0 2 0 】

このPKIディレクトリサービスにおいて、クライアント（証明書の失効情報を照会するアプリケーション）の数が多数になると、問い合わせの負荷の増大でシステムが破綻してしまうおそれがある。そのため、複数の分散されたPKIディレクトリサーバに大元のPKIディレクトリサーバの複製を作り、問い合わせの負荷の分散を図る方法がとられる。しかしながら、複製を作った場合には、それらを最新の情報に更新管理する必要があるが生じるが、複製の数の規模が非常に大きくなると、その更新管理が難しくなるという問題点があった。

【 0 0 2 1 】

したがって、この発明の目的は、PKIディレクトリサーバからPKIディレクトリクライアントへ公開鍵証明書の失効情報の通知を、効率よく行うことができる送信装置および送信方法、受信装置および受信方法、ならびに、送受信システムおよび送受信方法を提供することにある。

【 0 0 2 2 】

【課題を解決するための手段】

この発明は、上述した課題を解決するために、公開鍵証明情報を階層的に管理

するディレクトリの階層構造を送信する送信装置において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する管理手段と、管理手段によって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と、検出手段で検出された差分情報を送信する送信手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信装置である。

【 0 0 2 3 】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信する送信方法において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する管理のステップと、管理のステップによって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、検出のステップで検出された差分情報を送信する送信のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とを格納するようにしたことを特徴とする送信方法である。

【 0 0 2 4 】

また、この発明は、送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信装置において、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構

造の変化の差分からなる差分情報を受信する受信手段と、受信手段により受信された差分情報に基づき構成されるディレクトリの階層構造を管理する管理手段と、差分情報を選択的に取り込み、管理手段で管理されるディレクトリの階層構造を変更する変更手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されることを特徴とする受信装置である。

【 0 0 2 5 】

また、この発明は、送信された、公開鍵証明情報を階層的に管理するディレクトリの階層構造を受信する受信方法において、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構造の変化の差分からなる差分情報を受信する受信のステップと、受信のステップにより受信された差分情報に基づき構成されるディレクトリの階層構造を管理する管理のステップと、差分情報を選択的に取り込み、管理のステップで管理されるディレクトリの階層構造を変更する変更のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されることを特徴とする受信方法である。

【 0 0 2 6 】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信システムにおいて、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する第 1 の管理手段と、第 1 の管理手段によって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出手段と

、検出手段で検出された差分情報を送信する送信手段と、送信手段により送信された、差分情報を受信する受信手段と、受信手段により受信された差分情報に基づき構成されるディレクトリの階層構造を管理する第2の管理手段と、差分情報を選択的に取り込み、第2の管理手段で管理されるディレクトリの階層構造を変更する変更手段とを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信システムである。

【 0 0 2 7 】

また、この発明は、公開鍵証明情報を階層的に管理するディレクトリの階層構造を送信し、送信されたディレクトリの階層構造を受信する送受信方法において、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造を管理する第1の管理のステップと、第1の管理のステップによって管理されるディレクトリの階層構造の変化を検出し、検出結果に基づきディレクトリの階層構造の変化の差分からなる差分情報を求める検出のステップと、検出のステップで検出された差分情報を送信する送信のステップと、送信のステップにより送信された、差分情報を受信する受信のステップと、受信のステップにより受信された差分情報に基づき構成されるディレクトリの階層構造を管理する第2の管理のステップと、差分情報を選択的に取り込み、第2の管理のステップで管理されるディレクトリの階層構造を変更する変更のステップとを有し、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるようにしたことを特徴とする送受信方法である。

【 0 0 2 8 】

上述したように、請求項1および5に記載の発明は、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下にあつて、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階

層構造が管理され、管理されたディレクトリの階層構造の変化の検出結果に基づき求められた、ディレクトリの階層構造の変化の差分からなる差分情報が送信され、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されるため、受信側では、受信側が有している公開鍵証明情報が最新のものであるか否かを知ることができると共に、受信側が有している公開鍵情報が最新のものでないとした場合には、最新の公開鍵証明情報を取得することができる。

【 0 0 2 9 】

また、請求項 6 および 9 に記載の発明は、送信された、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出した検出結果に基づき求められたディレクトリの階層構造の変化の差分からなる差分情報を受信し、受信された差分情報に基づき構成されるディレクトリの階層構造が、選択的に取り込まれた差分情報に基づき変更されると共に、コンテナエントリおよび／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されているため、公開鍵証明情報が最新のものであるか否かを知ることができると共に、公開鍵情報が最新のものでないとした場合には、最新の公開鍵証明情報を取得することができる。

【 0 0 3 0 】

また、請求項 1 0 および 1 5 に記載の発明は、送信側では、自分の配下の情報を格納可能なコンテナエントリに認証局情報を対応付け、コンテナエントリの配下において、自分の配下の情報を格納できないリーフエントリにエンドエンティティ情報を対応付け、コンテナエントリとリーフエントリとからなるディレクトリの階層構造の変化を検出し、検出結果に基づき求められた、ディレクトリの階層構造の変化の差分からなる差分情報を送信し、受信側では、送信された差分情報を受信し、受信された差分情報に基づき構成されるディレクトリの階層構造が選択的に取り込まれた差分情報により変更されると共に、コンテナエントリおよ

び／またはリーフエントリには、最新の公開鍵証明情報を取得可能な情報と、最新の公開鍵証明情報の失効情報とが格納されて送信されるため、受信側では、受信側が有している公開鍵証明情報が最新のものであるか否かを知ることができると共に、受信側が有している公開鍵情報が最新のものでないとした場合には、最新の公開鍵証明情報を取得することができる。

【 0 0 3 1 】

【発明の実施の形態】

以下、この発明の実施の一形態について説明する。先ず、理解を容易とするために、この発明の実施の一形態の説明に先んじて、この発明に適用できるディレクトリサービスについて説明する。図 1 は、この発明に適用できる系の一例を示す。送信側 1 は、例えばインターネットや放送ネットワークなどの、図示されないネットワーク上に存在する多数のコンテンツをツリー状の階層構造に整理し、ディレクトリ構造として管理している。送信側 1 では、このディレクトリ構造を示すディレクトリ情報を放送ネットワーク 2 に対して送信する。受信側 3 は、図 2 に一例が示されるように、放送ネットワーク 2 に対して多数が接続され、放送ネットワーク 2 を介してなされた放送をそれぞれが受信可能とされている。受信側 3 は、放送ネットワーク 2 で放送されたディレクトリ情報を受信し、受信されたディレクトリ情報を参照して、放送ネットワーク 2 や他のネットワーク上に存在する多数の情報の中から必要な情報を選択し、入手することができる。

【 0 0 3 2 】

図 1 に一例が示されるように、送信側 1 は、送信側ディレクトリサービスクライアント 1 0（以下、送信側クライアント 1 0 と略す）、送信側ディレクトリサーバ 1 1（以下、送信側サーバ 1 1 と略す）および送信側ディレクトリサーバレプリケータ 1 2（以下、送信側レプリケータ 1 2 と略す）からなる。これら送信側クライアント 1 0、送信側サーバ 1 1 および送信側レプリケータ 1 2 は、互いに例えばインターネットといったネットワークなどで接続されており、相互に通信が行われる。

【 0 0 3 3 】

送信側クライアント 1 0 は、例えば図示されないネットワークなどによってコ

ンテンツを提供するコンテンツプロバイダであって、ディレクトリ構造の変更や更新を行う。送信側クライアント 1 0 は、ネットワーク上の何処に位置していてもよい。送信側サーバ 1 1 は、送信側クライアント 1 0 の内容照会や変更などを行い、ディレクトリ構造を管理する。送信側サーバ 1 1 は、ネットワーク上で分散して構成することができる。送信側レプリケータ 1 2 は、送信側サーバ 1 1 で管理されているディレクトリ構造をモニタしてディレクトリ構造の更新を検出する。そして、送信側レプリケータ 1 2 は、この検出結果に基づき、更新前の構造と更新後の構造とを比較して差分を抽出し、ディレクトリ構造の差分更新情報を構成する。差分更新情報は、放送ネットワーク 2 に送信される。差分更新情報の構成については、後述する。

【 0 0 3 4 】

受信側 3 は、受信側ディレクトリサーバレプリケータ 1 7（以下、受信側レプリケータ 1 7 と略す）、受信側ディレクトリサーバ 1 6（以下、受信側サーバ 1 6 と略す）および受信側ディレクトリサービスクライアント 1 5（以下、受信側クライアント 1 5 と略す）からなる。受信側 3 は、例えばパーソナルコンピュータや従来技術で述べた S T B、I R D などであり、受信側クライアント 1 5 は、ディレクトリ構造にアクセスして複数の異なる形式のデータの取得ならびに表示ができるようにされた、例えば W W W (World Wide Web) ブラウザなどのアプリケーションソフトウェアである。また、受信側サーバ 1 6 は、ローカルなデータベースからなり、ディレクトリ情報が格納される。

【 0 0 3 5 】

放送ネットワーク 2 で送信されたディレクトリ情報、ディレクトリ構造の更新情報および更新情報の差分情報などは、受信側レプリケータ 1 7 に受信される。受信側レプリケータ 1 7 では、受信されたこれらの情報に基づき、受信側サーバ 1 6 に格納されたデータベースを更新し、ディレクトリ構造の再構築を行う。受信側クライアント 1 5 は、例えばユーザの指示により、受信側レプリケータ 1 7 に対して必要な情報を要求する。受信側レプリケータ 1 7 は、この要求に基づき受信側サーバ 1 6 のデータベースを検索して、受信側クライアント 1 5 に対して、例えば要求された情報のアドレスを返す。受信側クライアント 1 5 は、このア

ドレスに基づき、例えば図示されないネットワーク上に存在する情報にアクセスすることができる。

【0036】

図3を用いて、ディレクトリ構造について説明する。ディレクトリは、図に示されるように、ツリー状の階層構造からなる。ツリーの各節点（ノード）をエントリと称し、各エントリには、情報が格納される。エントリには、ルートエントリ、コンテナエントリおよびリーフエントリの3種類が定義される。コンテナエントリは、さらに配下のエントリを包含することができるエントリである。コンテナエントリによって構成される階層を、以下、コンテナ階層と称する。

【0037】

コンテナエントリ以外のエントリを、リーフエントリと称する。リーフエントリは、配下にエントリを含むことができない、末端の節点である。リーフエントリによる階層を、以下、リーフ階層と称する。リーフ階層は、コンテナエントリの配下に構成される。

【0038】

また、ディレクトリツリーの最上位のエントリは、ルートエントリと称され、当該ディレクトリ構造で完結される世界全体を示すエントリである。なお、以下では、コンテナエントリは、少なくとも一つのリーフエントリあるいはコンテナエントリを配下に持つものとする。

【0039】

エントリは、複数の属性を持つ。エントリが持つ属性のうちで、ディレクトリツリーで一意に識別される名前を、エントリ名と称する。エントリ名によって、そのエントリの、ディレクトリ構造上の位置を特定することができる。図3の例では、ルートエントリには、エントリ名Aが与えられている。ルートエントリの直接的な配下のエントリには、図の左側に示されるリーフエントリにはエントリ名A、Bが、右側のコンテナエントリにはエントリ名A、Cがそれぞれ与えられる。以下、ルートエントリから階層構造を辿った順にピリオドで区切られて、各エントリに対してエントリ名が与えられる。

【0040】

図 4 は、コンテナエントリの構造の一例を示す。コンテナエントリは、コンテナエントリ自身の属性と、自分の配下のコンテナエントリおよびリーフエントリのリストとを有する。配下のエントリのリストは、要素を含まないこともできる。また、コンテナエントリ自身の属性は、図示されるように、複数持つことができる。コンテナエントリの属性は、図 4 B に示されるように、属性名と属性値とから構成される。

【 0 0 4 1 】

図 5 は、リーフエントリの構造の一例を示す。リーフエントリは、図 5 A に示されるように、リーフエントリの属性を複数、有する。図 5 B は、リーフエントリの属性のより具体的な例である。各属性は、属性名と属性値とからなる。例えばリーフエントリがコンテンツの検索情報である場合には、属性名の一つにアドレスがあり、その属性値として、インターネット上の場所を示す URL (Uniform Resource Locator) などの、コンテンツのアドレス情報が記述される。

【 0 0 4 2 】

ディレクトリ構造は、例えば、そのディレクトリ構造で完結する世界全体を表すルートエントリの下に、コンテナエントリが例えば情報ジャンルに応じてツリー状に分類され配されて、構成される。

【 0 0 4 3 】

送信側 1 の構成について、より具体的に説明する。送信側サーバ 1 1 には、図 3 ～図 5 で既に説明したような構成に準えて、ディレクトリ構造が管理されている。送信側クライアント 1 0 は、提供するコンテンツに応じて、送信側サーバ 1 1 で管理されているディレクトリ構造に対して変更などを加える。送信側サーバ 1 1 に対してなされた変更は、送信側レプリケータ 1 2 にモニタされる。

【 0 0 4 4 】

図 6 は、送信側レプリケータ 1 2 の機能を説明するための機能ブロック図である。送信側レプリケータ 1 2 は、例えば一般的なコンピュータシステムで構成することができ、CPU (Central Processing Unit)、メモリ、ハードディスクといった記録および記憶媒体、通信手段、タイマ、ユーザインターフェイスなどからなる。この図 6 に示される機能ブロックは、CPU 上で動作するアプリケーション

ョンソフトウェアにより実現され、各モジュールは、アプリケーションソフトウェア上の機能的な単位であって、それぞれがソフトウェアからなる。

【 0 0 4 5 】

送信側レプリケータ 1 2 は、更新検知モジュール 2 0、メッセージ生成モジュール 2 1 およびメッセージ放送モジュール 2 2 からなる。これらモジュール 2 0、2 1 および 2 2 のそれぞれは、コンテナ階層に関する処理を行うモジュールと、リーフ階層に関する処理を行うモジュールとを有する。

【 0 0 4 6 】

更新検知モジュール 2 0 は、送信側サーバ 1 1 を参照して、サーバ 1 1 上に管理されているディレクトリ構造に変化があったかどうかを検知するモジュールで、コンテナ階層更新検知モジュール 2 3 とリーフ階層更新検知モジュール 2 4 とからなる。コンテナ階層更新検知モジュール 2 3 は、送信側サーバ 1 1 をモニタして、コンテナ階層の構造の変化を検知する。また、リーフ階層更新検知モジュール 2 4 は、送信側サーバ 1 1 をモニタして、リーフ階層の構造およびリーフエントリの内容の変化を検知する。

【 0 0 4 7 】

メッセージ生成モジュール 2 1 は、更新検知モジュール 2 0 によるディレクトリ構造の変化の検知結果に基づく、ディレクトリ構造の差分更新情報が示されたメッセージを生成するモジュールで、コンテナストラクチャアップデートメッセージ生成モジュール 2 5 と、リーフエントリアップデートメッセージ生成モジュール 2 6 とからなる。コンテナストラクチャアップデートメッセージ生成モジュール 2 5 は、コンテナ階層更新検知モジュール 2 3 の検知結果に基づき、コンテナ階層における構造変化に関する差分更新情報が示されたメッセージを生成する。また、リーフエントリアップデートメッセージ生成モジュール 2 6 は、リーフ階層更新検知モジュール 2 4 の検知結果に基づき、リーフ階層における更新情報が示されたメッセージを生成する。

【 0 0 4 8 】

メッセージ放送モジュール 2 2 は、メッセージ生成モジュール 2 1 で生成されたメッセージを放送ネットワーク 2 に対して放送するモジュールで、コンテナス

トラクチャアップデートメッセージ放送モジュール 2 7 とリーフエントリアップデートメッセージ放送モジュール 2 8 とからなる。コンテナストラクチャアップデートメッセージ放送モジュール 2 7 は、上述した、コンテナストラクチャアップデートメッセージ生成モジュール 2 5 で生成されたメッセージを放送する。また、リーフエントリアップデートメッセージ放送モジュール 2 8 は、上述した、リーフエントリアップデートメッセージ生成モジュール 2 6 で生成されたメッセージを放送する。なお、メッセージ放送モジュール 2 2 からの放送ネットワーク 2 に対するメッセージの放送は、同一のメッセージが所定回数だけ、サイクリックに送信されて行われる。

【 0 0 4 9 】

次に、受信側 3 の構成について、より具体的に説明する。図 7 は、受信側クライアント 1 5 の機能を説明するための機能ブロック図である。受信側クライアント 1 5 は、例えば一般的なコンピュータシステムで構成することができ、CPU (Central Processing Unit)、メモリ、ハードディスクといった記録および記憶媒体、通信手段、ユーザインターフェイスなどからなる。この図 7 に示される機能ブロックは、CPU 上で動作するアプリケーションソフトウェアにより実現され、各モジュールは、アプリケーションソフトウェア上の機能的な単位であって、それぞれがソフトウェアからなる。

【 0 0 5 0 】

受信側クライアント 1 5 は、上述したように、例えば WWW ブラウザであり、供給されたコンテンツ、例えば画像データ、テキストデータ、音声データおよび動画データを統一的に表示および再生することができるようにされている。また、所定の入力手段を用いてユーザによって入力された指示に基づき、上述の表示ならびに再生を制御することができる。

【 0 0 5 1 】

受信側クライアント 1 5 は、ディレクトリ検索モジュール 3 0、ユーザ対話管理モジュール 3 1 およびコンテンツ取得モジュール 3 2 とからなる。また、ユーザ対話管理モジュール 3 1 に対して、例えばキーボードなどのテキスト入力手段、マウスなどのポインティングデバイスや表示装置などからなるユーザインター

フェイス 3 3 が接続される。ユーザの、受信側クライアント 1 5 に対するコンテンツ検索要求の入力は、ユーザインターフェイス 3 3 を用いて、ユーザ対話モジュール 3 1 に対して対話形式で行われる。

【 0 0 5 2 】

ユーザ対話モジュール 3 1 に対してコンテンツ検索要求が入力されると、ユーザ対話管理モジュール 3 1 からディレクトリ検索モジュール 3 0 に対して、コンテンツのアドレスを検索するため、コンテンツに対応するディレクトリエントリを検索するよう依頼が出される。ディレクトリ検索モジュール 3 0 では、この検索依頼に応じて、受信側サーバ 1 6 に対してディレクトリエントリ検索要求を送る。

【 0 0 5 3 】

受信側サーバ 1 6 での検索要求に基づくディレクトリエントリの検索結果がディレクトリ検索モジュール 3 0 に返される。検索結果は、ディレクトリ検索モジュール 3 0 からユーザ対話管理モジュール 3 1 に返される。そして、検索結果のディレクトリエントリ情報から、例えば検索結果がリーフエントリであれば、属性の一つであるコンテンツのアドレス情報が取り出される。ユーザ対話管理モジュール 3 1 からコンテンツ取得モジュール 3 2 に対して、取り出されたアドレス情報によって示されるコンテンツを取得するよう、コンテンツ取得依頼が出される。

【 0 0 5 4 】

コンテンツ取得モジュール 3 2 では、受け取ったコンテンツ取得依頼に基づき、コンテンツサーバ 3 5 に対してコンテンツの取得要求を送る。コンテンツサーバ 3 5 は、受信側クライアント 1 5 と、例えばインターネットといった双方向ネットワーク 3 6 を介して接続されるサーバであり、ユーザに対してコンテンツを提供する。コンテンツの提供は、双方向ネットワーク 3 6 を介して行ってもいいし、放送ネットワーク 2 によって行うこともできる。

【 0 0 5 5 】

コンテンツ取得要求に基づきコンテンツサーバ 3 5 から取得されたコンテンツは、例えば双方向ネットワーク 3 6 を介してコンテンツ取得モジュール 3 2 に供

給され、コンテンツ取得モジュール 3 2 からユーザ対話管理モジュール 3 1 に返される。ユーザ対話管理モジュール 3 1 では、受け取ったコンテンツをユーザインターフェイス 3 3 に出力する。

【 0 0 5 6 】

なお、要求するコンテンツが放送ネットワーク 2 で伝送されるものである場合には、コンテンツ取得モジュール 3 2 は、放送ネットワーク 2 で放送される所望のコンテンツを、コンテンツ取得依頼に基づき直接的に取得するようにしてもよい。

【 0 0 5 7 】

図 8 は、受信側サーバ 1 6 の機能を説明するための機能ブロック図である。この受信側サーバ 1 6 も、説明は省略するが、上述の受信側クライアント 1 5 と同様に、一般的なコンピュータシステムによって構成される。受信側サーバ 1 6 は、ディレクトリ更新要求処理モジュール 4 0、ディレクトリデータベース 4 1 およびディレクトリ検索要求処理モジュール 4 2 からなる。

【 0 0 5 8 】

ディレクトリデータベース 4 1 は、送信側サーバ 1 1 で管理されるディレクトリ構造に基づくディレクトリ情報が格納されている。上述したように、受信側レプリケータ 1 7 は、放送ネットワーク 2 を介して送信側 1 から送信されたディレクトリ構造の差分更新情報を受信する。詳細は後述するが、受信側レプリケータ 1 7 からディレクトリ更新要求処理モジュール 4 0 に対して、差分更新情報に基づきディレクトリデータベース 4 1 に格納されているディレクトリ情報の更新を行うよう、要求が出される。ディレクトリ更新要求処理モジュール 4 0 では、この要求に基づき、差分更新情報を用いてディレクトリデータベース 4 1 に格納されているディレクトリ情報の更新を行う。

【 0 0 5 9 】

一方、上述した受信側クライアント 1 5 からのディレクトリエントリの検索要求は、ディレクトリ検索要求処理モジュール 4 0 に受け取られる。ディレクトリ検索要求処理モジュール 4 0 によって、受け取った検索要求に基づきディレクトリデータベース 4 1 が検索される。検索した結果得られたディレクトリエントリ

、例えばリーフエントリ中のアドレス情報は、ディレクトリ検索要求処理モジュール42から受信側クライアント15に返される。

【0060】

上述したように系を構成することで、ユーザは、受信側クライアント15によってディレクトリ情報を検索し、検索結果として、所望のコンテンツが提供されるアドレス情報を得ることができる。そして、ユーザは、得られたアドレス情報に基づき所望のコンテンツを取得することができる。また、ディレクトリ構造は、常に送信側レプリケータ12によってモニタされ、ディレクトリ構造に変更が生じたときには、変更に伴うディレクトリ構造の差分更新情報が放送ネットワーク2を介して受信側レプリケータ17に供給される。ユーザ側では、供給された差分更新情報に基づき、受信側レプリケータ17によってディレクトリデータベース41に格納されたディレクトリ情報が変更される。そのため、ユーザは、常に実際のディレクトリ構造と同期したディレクトリ情報を、ディレクトリデータベース41に保持することができる。

【0061】

次に、図9および図10を用いて、ディレクトリ構造の変化に伴い生成される、ディレクトリ構造の差分更新情報について説明する。以下では、スキーマバージョンSvで特定されるあるコンテナ階層のコンテナエントリXの配下に、コンテナエントリCまたはリーフエントリ1を追加または削除する処理を、

(Sv, X, [+/-] [C/1])

と記述する。このディレクトリ構造に対する処理を示す記述は、この記述による処理で変化したディレクトリ構造の、元の構造との差分を表しており、これを差分更新情報として用いることができる。

【0062】

スキーマバージョンSvは、ディレクトリ構造の変更に伴い変化する値である。コンテナエントリX（またはC）は、コンテナエントリ名であり、ここでは大文字のアルファベットで表す。リーフエントリ1は、リーフエントリ名であり、ここでは小文字のアルファベットで表す。エントリの追加は「+」で表され、削除は「-」で表される。括弧「」中のスラッシュ記号は、その両側に記された何

方かが記述されることを示す。なお、図 9 および図 1 0 において、二重線の四角は、コンテナエントリを表し、単線の四角は、リーフエントリを表す。ルートエントリは、接続線だけが示され、本体は省略されている。

【 0 0 6 3 】

図 9 A において、図示されないルートエントリの配下にコンテナエントリ A のみが存在している。この状態をスキーマバージョン $S_v = 1$ とする。この状態に、上述の記述に従い $(1, A, +B)$ の処理を行う。すなわち、コンテナエントリ A の配下にコンテナエントリ B が加えられる。すると、図 9 B のようなディレクトリ構造になる。図 9 A の状態にコンテナエントリ B が加えられたことで、コンテナエントリの階層イメージが変化したとされ、スキーマバージョン $S_v = 2$ とされる。

【 0 0 6 4 】

図 9 B の状態に、さらに $(2, A, +a)$ の処理を行う。すなわち、コンテナエントリ A の配下にリーフエントリ a を加える。すると、図 9 C のようなディレクトリ構造になる。さらにまた、図 9 C の状態に $(2, A, -a)$ の処理を行う。すなわち、リーフエントリ a を、コンテナエントリ A の配下から削除する。すると、図 9 D のようなディレクトリ構造になる。さらに、図 9 D の状態に $(2, A, -B)$ の処理を行う。すなわち、コンテナエントリ B を、コンテナエントリ A の配下から削除する。すると、図 9 E のようなディレクトリ構造になる。

【 0 0 6 5 】

図 9 E の状態では、コンテナエントリの階層イメージが図 9 D の状態から変化しているため、スキーマバージョン S_v が更新され、スキーマバージョン $S_v = 3$ になる。したがって、図 9 E の状態において、コンテナエントリ A の配下にコンテナエントリ C を加える処理は、 $(3, A, +C)$ と記述できる。この処理を行うと、図 9 F に示されるディレクトリ構造となる。

【 0 0 6 6 】

この図 9 の例では、 $(1, A, +B)$ 、 $(2, A, +a)$ 、 $(2, A, -a)$ 、 $(2, A, -B)$ および $(3, A, +C)$ がそれぞれの段階での差分更新情報とされる。

【 0 0 6 7 】

図 1 0 は、ディレクトリ構造を変更する別の例を示す。上述の図 9 に示される例では、一度に一つの処理を行っていたが、図 1 0 では、2 つずつの処理をまとめて行っている。図 1 0 A は、図示されないルートエントリの配下にコンテナエントリ A のみが存在している。この状態がスキーマバージョン $S v = 1$ とする。この図 1 0 A の状態に対して、 $(1, A, +B)$ および $(1, A, +a)$ の処理を順に行う。すなわち、コンテナエントリ A の配下に、コンテナエントリ B とリーフエントリ a とが追加される。すると、図 1 0 B のようなディレクトリ構造となる。コンテナエントリの階層イメージが変わり、スキーマバージョン $S v = 2$ となる。

【 0 0 6 8 】

図 1 0 B の状態に、さらに、 $(2, A, -a)$ および $(2, B, +b)$ の処理を順に行う。すなわち、コンテナエントリ A の配下からリーフエントリ a を削除し、その後、コンテナエントリ B の配下にリーフエントリ b を追加する。すると、図 1 0 C のようなディレクトリ構造となる。

【 0 0 6 9 】

図 1 0 C の状態に対して、さらに、 $(2, B, +C)$ および $(2, C, +c)$ の処理を行う。すなわち、コンテナエントリ B の配下にコンテナエントリ C を追加した後に、追加されたコンテナエントリ C の配下にリーフエントリ c を追加する。この処理においては、追加されたコンテナエントリに対してさらにエントリが追加されるため、処理の前後を入れ換えることができない。処理後、図 1 0 D のようなディレクトリ構造となり、コンテナエントリの階層イメージが変わったため、スキーマバージョン $S v$ が更新され、スキーマバージョン $S v = 3$ とされる。

【 0 0 7 0 】

この図 1 0 の例では、 $(1, A, +B)$ および $(1, A, +a)$ 、 $(2, A, -a)$ および $(2, B, +b)$ 、ならびに、 $(2, B, +C)$ および $(2, C, +c)$ がそれぞれの段階での差分更新情報とされる。上述したように、複数の処理を一つのディレクトリ構造の更新処理としてまとめて行うときには、処理の順

序を考慮する必要がある。

【 0 0 7 1 】

なお、ディレクトリ構造の差分更新情報は、上述の例に限定されず、適用されるシステムに応じて様々な形式とすることができる。

【 0 0 7 2 】

また、リーフエントリに関しては、コンテナエントリの配下からの削除やコンテナエントリの配下への追加の他に、内容の修正だけが行われることがある。内容の修正だけが行われた場合には、ディレクトリ構造における変化は、生じない。この場合には、例えば、リーフエントリ名と、当該リーフエントリ中で修正のあった属性名および属性値の列で、差分更新情報が構成される。一例として、

```
{
  LeafEntryName,
  Set of { AttributeName, AttributeValue }
}
```

このように差分更新情報が記述される。

【 0 0 7 3 】

この発明が適用される系では、上述したように、差分更新情報は、送信側 1 から受信側 3 に対して放送ネットワーク 2 を介して一方向的に送信される。また、一つの送信側 1 に対して複数の受信側 3 が存在し、複数の受信側 3 の稼働状態もそれぞれ異なる。そのため、送信側 1 で管理されているディレクトリ情報と、受信側 3 で管理されているディレクトリ情報とを同期させる必要がある。

【 0 0 7 4 】

以下、送信側 1 の送信側サーバ 1 1 に格納されたディレクトリ情報と、受信側 3 の受信側サーバ 1 6 に格納されたディレクトリ情報とを同期させ、ディレクトリ構造の同期管理方法について説明する。

【 0 0 7 5 】

最初に、図 1 1 のフローチャートを用いてコンテナエントリの同期管理方法について説明する。まず、ステップ S 1 で、送信側クライアント 1 0 によって、送信側サーバ 1 1 で管理されているディレクトリ構造の、コンテナ階層の構成が変

更される。例えば、コンテナエントリの配下に新たなコンテナエントリやリーフエントリが追加される処理や、コンテナエントリの配下のコンテナエントリやリーフエントリが削除される処理が行われる。

【 0 0 7 6 】

次のステップ S 2 では、送信側サーバ 1 1 に対してなされた変更が送信側レプリケータ 1 2 によって検知され、検知結果に基づき、コンテナ階層構成の変更によるコンテナ構造更新情報 M s g . 1 が生成される。生成されたコンテナ構造更新情報 M s g . 1 は、放送ネットワーク 2 に対して放送される。コンテナ構造更新情報 M s g . 1 の放送は、同一の内容が所定回数、サイクリックに繰り返されて行われる。

【 0 0 7 7 】

放送されたコンテナ構造更新情報 M s g . 1 は、ステップ S 3 で、受信側レプリケータ 1 7 によって受信される。受信側レプリケータ 1 7 は、受信側サーバ 1 6 に格納されたディレクトリ情報に管理されるコンテナ階層構成を、受信したコンテナ構造更新情報 M s g . 1 に基づき変更する。これにより、送信側 1 と受信側 3 とで、ディレクトリ情報のコンテナ階層の構造の同期がとられる。

【 0 0 7 8 】

コンテナ構造更新情報 M s g . 1 のフォーマットは、例えば、
Container Structure Update Message {

MessageID,

差分更新情報

}

このように定義される。「MessageID」(メッセージID)は、このメッセージ(コンテナ構造更新情報 M s g . 1)の識別情報であって、例えば、このメッセージが生成される毎に 1 ずつ増加される整数である。また、「差分更新情報」は、コンテナ階層構成の変更による、上述したディレクトリ構造の差分更新情報である。

【 0 0 7 9 】

図 1 1 のフローチャートにおけるステップ S 2 の処理を、図 1 2 のフローチャ

ートを用いて、より詳細に説明する。この図 1 2 のフローチャートによる処理は、全て送信側レプリケータ 1 2 上で行われる。まず、ステップ S 1 0 で、送信側サーバ 1 1 上のコンテナエントリの階層構成の情報が全て読み込まれる。読み込まれたコンテナエントリの階層構成の情報は、送信側レプリケータ 1 2 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 1 として記憶される。

【 0 0 8 0 】

コピー 1 が記憶されたら、次のステップ S 1 1 で、タイマが所定の時間にセットされ、起動される。ステップ S 1 2 によって、タイマにセットされた所定時間を超過したかどうか判断され、所定時間を超過したと判断されたなら、処理はステップ S 1 3 に移行する。ステップ S 1 3 では、再び送信側サーバ 1 1 上のコンテナエントリの階層構成の情報が全て読み込まれる。読み込まれたコンテナエントリの階層構成は、送信側レプリケータ 1 2 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 2 として記憶される。

【 0 0 8 1 】

次のステップ S 1 4 では、ステップ S 1 0 で記憶されたコピー 1 と、ステップ S 1 3 で記憶されたコピー 2 とが比較される。比較の結果、両者に差分が無いとされれば（ステップ S 1 5）、処理はステップ S 1 1 へ戻され、再びタイマがセットされ、コピー 2 の記憶がなされる。

【 0 0 8 2 】

一方、ステップ S 1 4 で、コピー 1 とコピー 2 との間に差分があるとされれば、処理はステップ S 1 6 に移行する。ステップ S 1 6 では、コピー 1 とコピー 2 との差分に基づき、差分更新情報が生成される。そして、この差分更新情報が記述されたコンテナ構造更新情報 M s g . 1 が生成される。生成されたコンテナ構造更新情報 M s g . 1 は、放送ネットワーク 2 に対して送信され、放送される。放送されたコンテナ構造更新情報 M s g . 1 は、受信側レプリケータ 1 7 に受信される。

【 0 0 8 3 】

ステップ S 1 6 でコンテナ構造更新情報 M s g . 1 が放送されると、次のステ

ップS 1 7で、コピー 1 の内容がコピー 2 の内容で置き替えられ、処理はステップS 1 1に戻る。

【 0 0 8 4 】

図 1 1 のフローチャートにおけるステップS 3 の処理を、図 1 3 のフローチャートを用いて、より詳細に説明する。この図 1 3 のフローチャートによる処理は、全て受信側レプリケータ 1 7 上で行われる。最初のステップS 2 0 で、送信側レプリケータ 1 2 によって放送ネットワーク 2 を介して放送されたコンテナ構造更新情報M s g. 1 が、受信側レプリケータ 1 7 によって受信される。

【 0 0 8 5 】

ステップS 2 1 で、ステップS 2 0 での受信がコンテナ構造更新情報M s g. 1 の初回の受信かどうか判断される。そして、この受信が初回の受信では無いと判断されたら、処理はステップS 2 3 に移行し、受信されたコンテナ構造更新情報M s g. 1 のメッセージIDが受信側レプリケータ 1 7 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 3 として記憶される。

【 0 0 8 6 】

次のステップS 2 4 では、受信されたコンテナ構造更新情報M s g. 1 の内容、すなわち、コンテナ構造更新情報M s g. 1 に記述された差分更新情報に基づき、受信側サーバ 1 6 で管理されているディレクトリ情報が更新され、そのディレクトリ情報で示されるコンテナ階層の構成が変更される。ステップS 2 4 の処理の後、処理はステップS 2 0 に戻る。

【 0 0 8 7 】

一方、上述のステップS 2 1 で、ステップS 2 0 でのコンテナ構造更新情報M s g. 1 の受信が初回の受信では無いと判断されたら、処理はステップS 2 2 に移行する。ステップS 2 2 では、受信されたコンテナ構造更新情報M s g. 1 に記述されるメッセージIDと、前回の受信の際のステップS 2 3 の処理でコピー 3 として記憶されたメッセージIDとが同一であるかどうか判断される。若し、同一であるとされれば、処理はステップS 2 0 に戻る。

【 0 0 8 8 】

一方、ステップS22で、両者のメッセージIDが同一では無いとされれば、処理はステップS23に移行する。ステップS23では、上述したように、メッセージIDが記憶媒体にコピー3として記憶される。この場合には、新たに受信されたメッセージIDで、前回受信され記憶されたメッセージIDが例えば上書きされることになる。そして、次のステップS24で、受信されたコンテナ構造更新情報Msg. 1に基づき、受信側サーバ16上のコンテナエントリ階層の内容が変更される。

【0089】

次に、図14のフローチャートを用いてリーフエントリの同期管理方法について説明する。まず、ステップS30で、送信側クライアント10によって、送信側サーバ11で管理されているディレクトリ構造の、あるコンテナエントリの配下のリーフエントリが変更される。例えば、あるコンテナエントリの配下の新たなリーフエントリの追加や、あるコンテナエントリの配下のリーフエントリの削除や修正といった処理が行われる。

【0090】

次のステップS31では、送信側サーバ11のあるコンテナエントリの配下のリーフエントリに対してなされた変更が送信側レプリケータ12によって検知される。そして、検知結果に基づき、あるコンテナエントリ配下のリーフエントリの変更によるリーフ更新情報Msg. x1が生成される。生成されたリーフ更新情報Msg. x1は、放送ネットワーク2を介して複数の受信側レプリケータ17に対してサイクリックに放送される。

【0091】

放送されたリーフ更新情報Msg. x1は、ステップS32で、受信側レプリケータ17によって受信される。受信側レプリケータ17は、受信したリーフ更新情報Msg. x1に基づき、受信側サーバ16に格納されたディレクトリ情報に管理される、対応するリーフエントリを変更する。これにより、送信側1と受信側3とで、ディレクトリ情報のリーフエントリの同期がとられる。

【0092】

リーフ更新情報Msg. x1のフォーマットは、例えば、

Leaf Entry Update Message {

MessageID,

差分更新情報

}

このように定義される。「MessageID」(メッセージID)は、このメッセージ(リーフ更新情報Msg. x1)の識別情報であって、例えば、このメッセージが生成される毎に1ずつ増加される整数である。また、「差分更新情報」は、上述したディレクトリ構造の差分更新情報である。

【0093】

図14のフローチャートにおけるステップS31の処理を、図15のフローチャートを用いて、より詳細に説明する。この図15のフローチャートによる処理は、全て送信側レプリケータ12上で行われる。先ず、ステップS40で、送信側サーバ11上の、あるコンテナエントリの配下のリーフエントリ名が全て読み込まれる。読み込まれたリーフエントリ名は、送信側レプリケータ12が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー4として記憶される。

【0094】

コピー1が記憶されたら、次のステップS41で、タイマが所定の時間にセットされ、起動される。ステップS42によって、タイマにセットされた所定時間を超過したかどうか判断され、所定時間を超過したと判断されたなら、処理はステップS43に移行する。ステップS43では、再び送信側サーバ11上の、あるコンテナエントリの配下のリーフエントリ名が全て読み込まれる。読み込まれたリーフエントリ名は、送信側レプリケータ12が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー5として記憶される。

【0095】

次のステップS44では、ステップS40で記憶されたコピー4と、ステップS43で記憶されたコピー5とが比較される。比較の結果、両者の間に差分が無いとされれば(ステップS45)、処理はステップS41へ戻され、再びタイマがセットされ、コピー5の記憶が行われる。

【 0 0 9 6 】

一方、ステップ S 4 4 で、コピー 4 とコピー 5 との間に差分があるとされれば、処理はステップ S 4 6 に移行する。ステップ S 4 6 では、コピー 4 とコピー 5 との差分に基づき、差分更新情報が生成される。そして、この差分更新情報が記述されたリーフ更新情報 M s g . x 1 が生成される。生成されたリーフ更新情報 M s g . x 1 は、放送ネットワーク 2 に対して送信され、放送される。放送されたリーフ更新情報 M s g . x 1 は、複数の受信側レプリケータ 1 7 に受信される。

【 0 0 9 7 】

ステップ S 4 6 でリーフ更新情報 M s g . x 1 が放送されると、次のステップ S 4 7 で、コピー 4 の内容がコピー 5 の内容で書き替えられ、処理はステップ S 4 1 に戻される。

【 0 0 9 8 】

なお、上述の図 1 5 のフローチャートの処理は、送信側レプリケータ 1 2 によって、送信側サーバ 1 1 が管理するディレクトリ構造上の全てのコンテナエントリに対して行われる。

【 0 0 9 9 】

図 1 4 のフローチャートにおけるステップ S 3 2 の処理を、図 1 6 のフローチャートを用いて、より詳細に説明する。この図 1 6 のフローチャートによる処理は、全て受信側レプリケータ 1 7 上で行われる。最初のステップ S 5 0 で、送信側レプリケータ 1 2 によって放送ネットワーク 2 を介して放送されたリーフ更新情報 M s g . x 1 が、受信側レプリケータ 1 7 によって受信される。

【 0 1 0 0 】

ステップ S 5 1 で、ステップ S 5 0 での受信がリーフ更新情報 M s g . x 1 の初回の受信であるかどうか判断される。若し、この受信が初回の受信で無いと判断されたら、処理はステップ S 5 3 に移行し、受信されたリーフ更新情報 M s g . x 1 のメッセージ I D が受信側レプリケータ 1 7 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 6 として記憶される。

【 0 1 0 1 】

次のステップ S 5 4 では、受信されたリーフ更新情報 M s g . x 1 の内容、すなわち、リーフ更新情報 M s g . x 1 に記述された差分更新情報に基づき、受信側サーバ 1 6 で管理されているディレクトリ情報のうち、対応するリーフエントリの内容が変更される。ステップ S 5 4 の処理の後、処理はステップ S 5 0 に戻される。

【 0 1 0 2 】

一方、上述のステップ S 5 1 で、ステップ S 5 0 でのリーフ更新情報 M s g . x 1 の受信が初回の受信では無いと判断されたら、処理はステップ S 5 2 に移行する。ステップ S 5 2 では、受信されたリーフ更新情報 M s g . x 1 に記述されるメッセージ I D と、前回の受信の際のステップ S 5 3 の処理でコピー 6 として記憶されたメッセージ I D とが同一であるかどうか判断される。若し、同一であるとされれば、処理はステップ S 5 0 に戻される。

【 0 1 0 3 】

一方、ステップ S 5 2 で、両者のメッセージ I D が異なるとされれば、処理はステップ S 5 3 に移行する。ステップ S 5 3 では、上述したように、メッセージ I D が記憶媒体にコピー 6 として記憶される。この場合には、新たに受信されたメッセージ I D で、前回受信され記憶されたメッセージ I D が例えば上書きされることになる。そして、次のステップ S 5 4 で、受信されたリーフ更新情報 M s g . x 1 に基づき、受信側サーバ 1 6 上の対応するリーフエントリの内容が変更される。

【 0 1 0 4 】

ところで、上述したように、図 1 4 のフローチャート中のステップ S 3 1 による、リーフ更新情報 M s g . x 1 の放送は、送信側 1 で管理されるディレクトリ構造中の、全テナエントリのそれぞれに関して行われ、放送されるリーフ更新情報 M s g . x 1 は、膨大な量になることが予想される。したがって、従来技術で問題点として既に述べたように、受信側 3 において全てのリーフ更新情報 M s g . x 1 を受信し、図 1 6 のフローチャートによる処理を行うことは、大きな負担になる。そのため、受信側 3 では、必要とされている、すなわち、頻繁に照会されるテナエントリの配下のリーフエントリに対するリーフ更新情報 M s

g. x 1 のみを、放送された多数のリーフ更新情報 M s g. x 1 から、効率よくフィルタ処理する必要がある。

【 0 1 0 5 】

例えば、受信側レプリケータ 1 7 が家庭内でテレビジョン受像機などに接続して用いられる、セットトップボックス (S T B) のような、処理能力や記憶容量が限定された、すなわち、コンピュータリソースに制約のある環境に実装される場合を想定する。この場合には、放送されるリーフ更新情報 M s g. x 1 の取得には限度がある。そのため、受信されたリーフ更新情報 M s g. x 1 を取捨選択して装置内に取り込み、記憶コストやメッセージ処理コストの軽減を図る必要がある。すなわち、受信側レプリケータ 1 7 において、不必要な記憶や処理にかかるコストを制限する必要がある。特に、ディレクトリサービスが普及し、送信側サーバ 1 1 で管理される対象のディレクトリ構造が巨大になるのに伴い、上述のリーフ更新情報 M s g. x 1 の取捨選択は、より重要な事項となってくる。

【 0 1 0 6 】

以下に、リーフ更新情報 M s g. x 1 に対してフィルタ処理を行う方法について説明し、さらに、この発明の主旨に係る、フィルタ処理を効率的に行う方法について説明する。送信側レプリケータ 1 2 は、放送されるリーフ更新情報 M s g. x 1 に対して、受信側レプリケータ 1 7 においてフィルタ処理を行うためのフィルタリングマスクを付加する。フィルタリングマスクを解釈するためのマスクスキーマ構造と、マスクスキーマ構造を送信側レプリケータ 1 2 から受信側レプリケータ 1 7 に通知する方法などについては、後述する。

【 0 1 0 7 】

リーフ更新情報 M s g. x 1 にフィルタリングマスクを付加したメッセージ (M s g. x 1 ') の構造を、次のように定義し、上述のリーフ更新情報 M s g. x 1 を全てこのリーフ更新情報 M s g. x 1 ' で置き替える。すなわち、リーフ更新情報 M s g. x 1 ' は、

```
Leaf Entry Update Message {
    MessageID,
    FilteringMask,
```

差分更新情報

}

このように定義される。「MessageID」（メッセージID）は、上述のリーフ更新情報Msg. x 1の場合と同様に、このメッセージ（リーフ更新情報Msg. x 1'）の識別情報であって、例えば、このメッセージが生成される毎に1ずつ増加される整数である。「差分更新情報」は、ここに記述されるフィルタリングマスクで特定されるコンテナエントリ配下のリーフエントリの、追加、削除および属性変更といった手続の情報である。

【0108】

「FilteringMask」（フィルタリングマスク）の構造は、次のように定義される。すなわち、フィルタリングマスクは、

```
FilteringMask {
    MaskSchema Version,
    Mask Value
}
```

このように定義される。「MaskSchema Version」（マスクスキーマバージョン）は、例えば上述のコンテナ構造更新情報Msg. 1におけるメッセージIDに相当し、例えばこのフィルタリングマスクが生成される毎に1、増加される値である。「Mask Value」（マスク値）は、例えばビット列あるいはバイト単位で表されるマスクの値である。

【0109】

なお、マスク値の構造は、マスクスキーマバージョンによって対応付けられるマスクスキーマ（後述する）によって規定される。マスクスキーマは、後述する別のメッセージによって送信側レプリケータ12から受信側レプリケータ17に通知される。

【0110】

マスク値の割当方法について説明する。この実施の一形態では、あるコンテナエントリの配下のコンテナエントリのそれぞれを、所定ビット数からなるビット列で識別する。受信側レプリケータ17では、受信されたリーフ更新情報Msg

． x 1' 中に記述されるマスク値を参照することによってフィルタ処理を行い、必要なリーフ更新情報 M s g . x 1' を選択的に抽出することができる。

【 0 1 1 1 】

フィルタリングマスクのマスク値のビット配列構造は、コンテナエントリの階層構造に対応させて決定される。例えば図 1 7 A に一例が示されるように、図 3 で説明したエントリ名の記述方法に倣い、上位のコンテナエントリ X の配下のエントリ X . A、X . B、X . C、X . D および X . E を互いに識別するために、それぞれ 3 ビットのマスク値 (0 0 0)、(0 0 1)、(0 1 0)、(0 1 1) および (1 0 0) が割り当てられる。なお、[...] は、より上位のコンテナエントリが存在することを示す。

【 0 1 1 2 】

このようにマスク値が与えられた、コンテナエントリ X の配下のコンテナエントリに対して、エントリの追加や削除が行われた場合、図 1 8 のフローチャートに従って処理が行われ、コンテナエントリの増減に応じてマスク値の割り当てを行う。なお、以下では、コンテナエントリの追加や削除が行われる前のコンテナ階層を、更新前コンテナ階層と称する。更新前コンテナ階層のマスク桁数 M' は、送信側レプリケータ 1 2 の例えばメモリに記憶されているものとする。

【 0 1 1 3 】

まず、最初のステップ S 6 0 で、送信側レプリケータ 1 2 によって、対象となるコンテナエントリの配下のコンテナエントリの数 N が取得される。コンテナエントリ中の、配下のコンテナエントリのリストを参照することで、コンテナエントリ数 N が求められる。次のステップ S 6 1 では、N 個の要素を一意に識別可能なビット数 M が選ばれ、マスクの桁数が M 桁とされる。例えば、上述した図 1 7 A の例では、コンテナエントリ X は、配下のコンテナエントリを 5 個有しているので、5 個を一意に識別可能なビット数 [3] がマスクの桁数とされる。

【 0 1 1 4 】

次に、ステップ S 6 2 で、上述のステップ S 6 1 で割り当てられたビット数 M が、対応する更新前コンテナ階層に割り当てられたマスク桁数 M' と同一かどうか判断される。若し、マスク桁数 M とマスク桁数 M' とが同一であれば、処理

はステップ S 6 3 に移行する。

【 0 1 1 5 】

ステップ S 6 3 では、更新後のコンテナ階層のコンテナエントリのうち、更新前コンテナ階層のコンテナエントリに対応するエントリには、同一のマスク値を割り当てる。さらに、次のステップ S 6 4 で、例えば更新後のコンテナ階層に新規にコンテナエントリの追加が起こったなどして、更新後のコンテナ階層に、更新前コンテナ階層に対応するコンテナエントリが存在しないコンテナエントリがあった場合、そのコンテナエントリに対して、同一のコンテナ階層の他のコンテナエントリのマスク値と重複しないようなマスク値が与えられる。

【 0 1 1 6 】

一方、上述のステップ S 6 2 で、マスク桁数 M とマスク桁数 M' とが同一ではないと判断されたら、処理はステップ S 6 5 に移行し、当該コンテナ階層の全コンテナエントリのそれぞれに対して、一意にマスク値が与えられる。

【 0 1 1 7 】

例えば、上述の図 1 7 A の状態に、新たにコンテナエントリ " ... X. F " が追加され、図 1 7 B のようなコンテナ階層になったとする。コンテナエントリ " ... X " の配下のコンテナエントリ数 N は、6 であり、一意に表現するためには 3 ビットが必要とされ、更新後のコンテナエントリ " ... X " の配下のコンテナ階層のマスク桁数 M = 3 である。更新前コンテナ階層のマスク桁数 M' = 3 であって、マスク桁数 M' とマスク桁数 M とは等しい。したがって、図 1 7 B に示される各コンテナエントリ " ... X. A " 、 " ... X. B " 、 " ... X. C " 、 " ... X. D " および " ... X. E " は、更新前コンテナ階層の対応するエントリのマスク値がそれぞれ割り当てられる（ステップ S 6 3）。一方、新規に追加されたコンテナエントリ " ... X. F " は、同じコンテナ階層の他のコンテナエントリと重複しないように、マスク値 (1 0 1) が割り当てられる（ステップ S 6 4）。

【 0 1 1 8 】

また例えば、上述の図 1 7 A の状態から、コンテナエントリ " ... X. C " を削除して、図 1 7 C のようなコンテナ階層になったとする。この場合、コンテナ

エントリ” ... X” 配下のコンテナエントリ数 N は、4であり、2ビットのマスク桁数 M で各コンテナエントリを識別可能なので、更新後のコンテナ階層のマスク桁数 $M=2$ である。一方、更新前コンテナ階層のマスク桁数 $M'=3$ であって、更新前と更新後とで、マスク桁数が異なる。したがって、ステップS65の処理によって、当該階層の全エントリに、マスク桁数 $M=2$ で新たにマスク値が割り当てられる。

【0119】

さらに、図17Cの状態に、新たにコンテナエントリ” ... X. G” が追加され、図17Dの状態になったとする。この場合、コンテナエントリ” ... X” 配下のコンテナエントリ数 N は5であり、コンテナエントリを互いに識別するためには、マスク桁数 $M=3$ とする必要がある。マスク桁数 M が更新前のマスク桁数 $M'=2$ と異なるため、ステップS65の処理によって、コンテナエントリ” ... X” の配下の全コンテナエントリに、新たにマスク値が割り当てられる。

【0120】

マスク値のビットアサインは、ディレクトリ構造の上位側から、コンテナ階層順にシリアルになされる。一方、この実施の一形態では、上述のように、同一階層に存在するエントリ数によってマスク桁数が異なる。また、エントリの削除や追加などによって、コンテナ階層中のエントリ数が変化し、それに伴いマスク桁数が変化する。そのため、マスク値を表すビット列中のどのビットがどのコンテナエントリ（あるいはコンテナ階層）に対応するかを判断し、マスク値を解釈するための情報機構が必要となる。

【0121】

この実施の一形態では、マスク値を解釈するための情報機構として、次に示すマスキーマ(MaskSchema)を定義する。マスキーマは、

```
MaskSchema {
    MaskSchema Version,
    TotalMaskLength,
    Set of ContainerEntryMaskSchema
}
```

このように定義される。「MaskSchema Version」(マスクスキーマバージョン)は、例えば上述のコンテナ構造更新情報Msg. 1におけるメッセージIDに相当し、例えば対応するフィルタリングマスクが生成される毎に1、増加される値である。「TotalMaskLength」(全マスク長)は、全体のコンテナ階層に対応する、マスク値全体のビット長を表す。すなわち、全マスク長は、ディレクトリ構造の全ての階層を表現するために必要なビット数に対応する。「Set of ContainerEntryMaskSchema」(セットオブコンテナエントリマスクスキーマ)は、後述する「ContainerEntryMaskSchema」(コンテナエントリマスクスキーマ)の配列を表す。

【 0 1 2 2 】

上述のコンテナエントリマスクスキーマは、あるコンテナエントリに対応するフィルタリングマスクを規定する。すなわち、コンテナエントリマスクスキーマは、

```
ContainerEntryMaskSchema {
    ContainerEntryName,
    OffsetLength,
    MaskLength,
    AssignedMaskValue
}
```

このように定義される。「ContainerEntryName」(コンテナエントリ名)は、対象となるコンテナエントリのエントリ名を表す文字列である。「OffsetLength」(オフセット長)は、このコンテナエントリに対応するフィルタリングマスクの、全マスク値の最初のビットからのオフセット値であり、「MaskLength」(マスク長)は、マスク値の桁数(ビット長)である。「AssignedMaskValue」(割り当てマスク値)は、対象となるコンテナエントリに割り当てられたマスク値であり、ビット列で表される。

【 0 1 2 3 】

図19を用いて、コンテナエントリマスクスキーマの符号化について説明する。図19Aは、上述の図17Aに対応する図であって、上位のコンテナエントリ

” ... X” の配下に、コンテナエントリ名” ... X. A”、” ... X. B”、” ... X. C”、” ... X. D” および” ... X. E” の 5 つのコンテナエントリが存在し、それぞれ 3 桁のマスク長で以てマスク値が割り当てられている。なお、ここでは説明のため、これら 5 つのコンテナエントリは、配下に他のエントリを有しないものとする。

【 0 1 2 4 】

図 1 9 B は、コンテナエントリ” ... X. C” のマスク値の一例を示す。この例では、オフセット長が 7 7 ビットであることから、コンテナエントリ” ... X. C” に 3 ビットのマスク長で割り当てられた割り当てマスク値が、コンテナエントリ” ... X. C” のマスク値の 7 8 ビット目から開始される 3 ビットであることが分かる。オフセット長に含まれる 7 7 ビットのマスク値は、コンテナエントリ” ... X. C” より上位のコンテナエントリに対応する割り当てマスク値である。

【 0 1 2 5 】

このように、マスク値における対象コンテナエントリの割り当てマスク値の位置が規定され、コンテナエントリマスクスキーマが符号化される。

【 0 1 2 6 】

コンテナエントリマスクスキーマのより具体的な例を示す。上述したコンテナエントリ” ... X. C” に対応するコンテナエントリマスクスキーマは、例えば

```
ContainerEntryMaskSchema {
    "...X.C", (ContainerEntryName)
    77,      (OffsetLength)
    3,       (MaskLength)
    010      (AssignedMaskValue)
}
```

このようになる。なお、括弧()内は、説明のためのものであって、実際に記述する必要は無い。

【 0 1 2 7 】

また、図 1 9 A に示されるコンテナエントリ "... X. D" に対応するコンテナエントリマスクスキーマは、例えば、

```
ContainerEntryMaskSchema {
    "...X.D",
    77,
    3,
    011
}
```

このようになる。

【 0 1 2 8 】

このときのマスクスキーマは、例えばマスクスキーマバージョンを 4 9 8、全マスク長を 1 3 4 ビットとした場合、

```
MaskSchema {
    498,    (MaskSchema Version)
    134,    (TotalMaskLength)
```

.....

```
ContainerEntryMaskSchema {
    "...X.C",
    77,
    3,
    010
}
```

```
ContainerEntryMaskSchema {
    "...X.D",
    77,
    3,
    011
}
```

.....

}

このようになる。上述の例では、コンテナエントリ” ... X. Cおよび” ... X. Dのコンテナエントリマスクスキーマがマスクスキーマ中に記述されているが、[....]の部分には、さらに他のコンテナエントリマスクスキーマが記述される。この例で分かるように、マスクスキーマには、一つのディレクトリ構造における全コンテナエントリに関するコンテナエントリマスクスキーマが記述される。

【 0 1 2 9 】

なお、この例で、全マスク長が134ビットとなっているのに対して、コンテナエントリ” ... X. Cおよび” ... X. Dについてのコンテナエントリマスクスキーマでは、オフセット値が77ビットおよびマスク長が3ビットの、合計で80ビットである。これは、これらコンテナエントリ” ... X. Cおよび” ... X. Dの配下にも、さらにコンテナ階層が存在することを示している。

【 0 1 3 0 】

上述したマスクスキーマにおいて、コンテナエントリ” ... X. Cに対応するフィルタリングマスクの符号化は、例えば、

```
FilteringMask {
    498,    (MaskSchema Version)
    .....010    (Mask Value)
}
```

このようになる。なお、マスク値(Mask Value)は、[011]以外の部分も全て、他の階層のコンテナエントリの割り当てマスク値からなるビットで埋められる。

【 0 1 3 1 】

同様に、コンテナエントリ” ... X. Dに対応するフィルタリングマスクの符号化は、例えば、

```
FilteringMask {
    498,    (MaskSchema Version)
    .....011    (Mask Value)
```

}

このようになる。

【 0 1 3 2 】

送信側レプリケータ 1 2 では、送信側サーバ 1 1 をモニタして、コンテナエントリの階層構造の変更を検知して、上述したマスキスキーマの変更を行う。したがって、受信側 3 において適切なフィルタ処理を行うためには、送信側レプリケータ 1 2 によって、階層構造の変更に基づく差分更新情報の通知と共に、変更されたマスキスキーマが受信側レプリケータ 1 7 に通知される必要がある。

【 0 1 3 3 】

この発明では、マスキスキーマを送信側レプリケータ 1 2 から受信側レプリケータ 1 7 に通知するために、上述したコンテナ構造更新情報 M s g . 1 の構造に対して、マスキスキーマ構造を追加する。マスキスキーマ構造を追加されたコンテナ構造更新情報 M s g . 1 ' を、

Container Structure Update Message {

MessageID,

差分更新情報,

MaskSchema

}

このように定義する。マスキスキーマは、コンテナ階層の構成が変更される毎に変更される可能性がある。そのため、このコンテナ構造更新情報 M s g . 1 ' も、コンテナ階層の構成の変更に応じて生成される。「MessageID」（メッセージ ID）は、コンテナ構造更新情報 M s g . 1 ' が生成される毎に 1 ずつ増加される整数である。以下では、上述したコンテナ構造更新情報 M s g . 1 を、全てこのコンテナ構造更新情報 M s g . 1 ' に置き替えるものとする。

【 0 1 3 4 】

送信側レプリケータ 1 2 では、上述した図 1 5 のフローチャートにおけるステップ S 4 6 で、コンテナ階層に対応させたフィルタリングマスクが付加されたメッセージである、リーフ更新情報 M s g . x 1 ' を生成し、受信側レプリケータ 1 7 に放送している。ここで、受信側 3 では、リーフ更新情報 M s g . x 1 ' に

よる受信側レプリケータ 1 7 でのフィルタ処理を行う前に、受信側クライアント 1 5 が必要としているコンテナ階層の対象部分を特定しておく必要がある。

【 0 1 3 5 】

この実施の一形態では、受信側レプリケータ 1 7 において、対象となるコンテナ階層をフィルタ処理するためのマスクをリストにした、ターゲットマスクリストを作成する。

【 0 1 3 6 】

図 2 0 を用いてターゲットマスクリストについて説明する。まず、図 2 0 A に示されるようなディレクトリ構造を想定する。図 2 0 A のディレクトリ階層は、最上位のルートエントリ以外は全てコンテナエントリで構成されているものとする。各四角はコンテナエントリを表し、二重線の四角で示されるコンテナエントリは、例えばユーザの嗜好に基づき受信側クライアント 1 5 でフィルタ処理を行うように特定されたエントリである。各エントリ内に表示された数字は、当該エントリ毎に割り当てられたマスク値である。

【 0 1 3 7 】

図 2 0 A に示されるように、ユーザの嗜好に基づきフィルタ処理するように受信側クライアント 1 5 で特定されたコンテナエントリのそれぞれに対して、マスク 1 ～ 5 が割り当てられている。このディレクトリ構造において、マスク 1 ～ 5 の全マスク長分のマスク値は、ディレクトリ構造を上位側から辿り、マスク 1 が [0 0 0] 、マスク 2 が [0 0 1 0] マスク 3 が [0 1 0] 、マスク 4 が [1 0 0 0 0] およびマスク 5 が [1 0 0 1 0] となる。

【 0 1 3 8 】

図 2 0 B は、このように特定されたマスクがリストとされたターゲットマスクリストの一例を示す。ターゲットマスクリストは、ディレクトリの構造を特定するスキーマバージョンと、ユーザの嗜好などに基づき受信側クライアント 1 5 で特定された上述したマスク値のリストとからなる。すなわち、このターゲットマスクリストは、スキーマバージョンで記述されたディレクトリ構造でのみ有効なリストである。

【 0 1 3 9 】

図 2 1 は、ターゲットマスキリストを作成する処理のフローチャートである。このフローチャートは、受信側レプリケータ 1 7 で実行される。まず、最初のステップ S 7 0 で、受信側レプリケータ 1 7 によって、コンテナ構造更新情報 M s g . 1 ' が受信される。ステップ S 7 1 で、ステップ S 7 0 での受信がコンテナ構造更新情報 M s g . 1 ' の初回の受信であるかどうか判断され、初回の受信であると判断されれば、処理はステップ S 7 3 に移行する。

【 0 1 4 0 】

ステップ S 7 3 では、受信されたコンテナ構造更新情報 M s g . 1 ' のメッセージ ID を、受信側レプリケータ 1 7 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 7 として記憶される。

【 0 1 4 1 】

次のステップ S 7 4 で、受信されたコンテナ構造更新情報 M s g . 1 ' の内容に基づき、コンテナ階層を生成する。生成されたコンテナ階層を示す情報が受信側レプリケータ 1 7 から受信側クライアント 1 5 に提示され、特定すべきコンテナエントリの選択が促される。例えば、受信側クライアント 1 5 では、所定の表示手段を用いて、供給されたコンテナ階層を示す情報に基づく表示を行う。ユーザは、この表示に基づき、所定の方法で必要なコンテナエントリを選択する。選択されたコンテナエントリの情報は、受信側クライアント 1 5 から受信側レプリケータ 1 7 に渡される。

【 0 1 4 2 】

なお、コンテナエントリの特定は、ユーザの直接的な選択によってなされるのに限定されない。例えば、受信側クライアント 1 5 によって、ユーザが照会を行ったコンテナエントリの情報を蓄積し、蓄積された情報に基づきユーザの嗜好の傾向を学習して自動的に必要と思われるコンテナエントリを選択するようにもできる。さらに、ユーザによる直接的な選択と、学習による自動的な選択とを併用することもできる。

【 0 1 4 3 】

このようにして、ステップ S 7 4 でコンテナエントリが選択されると、ステップ S 7 5 で、選択されたコンテナ階層に対応するフィルタリングマスクが設定さ

れる。設定されたフィルタリングマスクの一覧は、ターゲットマスクリストとして、例えば受信側レプリケータ17が有する、例えばメモリやハードディスクといった記録または記憶媒体に記憶される。

【0144】

一方、上述のステップS71で、コンテナ構造更新情報Msg. 1'の受信が初回ではないと判断されれば、処理はステップS72に移行する。ステップS72では、受信されたコンテナ構造更新情報Msg. 1'のメッセージIDが、前回までのコンテナ構造更新情報Msg. 1'の受信によりステップS73でコピー7として記憶媒体に記憶されたメッセージIDと同一かどうか判断される。

【0145】

若し、両者が同一であると判断されたら、処理はステップS70に戻る。一方、ステップS73で両者が同一では無いと判断されたら、処理はステップS74に移行し、今回受信されたコンテナ構造更新情報Msg. 1'のメッセージIDが前回までのメッセージIDの代わりに記憶媒体に記憶され、今回受信されたコンテナ構造更新情報Msg. 1'に基づき以降の処理が行われる。

【0146】

図22は、図21のフローチャートに従って作成されたターゲットマスクリストに基づき、放送されたリーフ更新情報Msg. x1'を選択的に受信する処理を示すフローチャートである。受信側レプリケータ17は、ターゲットマスクリストに列挙されたフィルタリングマスクを有するリーフ更新情報Msg. x1'を、放送ネットワーク2で放送されたリーフ更新情報Msg. x1'の中から選択的に受信する。選択的に受信されたリーフ更新情報Msg. x1'により、上述した図14のフローチャートにおけるステップS32の処理が実行される。

【0147】

図22において、まず、最初のステップS80で、受信側レプリケータ17によって、放送ネットワーク2によって放送されたリーフ更新情報Msg. x1'が受信される。受信側レプリケータ17では、記憶媒体に記憶されているターゲットマスクリストが参照され、受信されたリーフ更新情報Msg. x1'に示されるフィルタリングマスクがターゲットマスクリストに存在するかどうか判断

される。若し、ターゲットマスクリスト中に存在しなければ、処理はステップ S 80 に戻される。

【0148】

一方、ステップ S 81 で、当該フィルタリングマスクがターゲットマスクリスト中に存在すると判断されれば、処理はステップ S 82 に移行し、ステップ S 80 での受信がリーフ更新情報 M s g. x 1' の初回の受信であるかどうか判断される。若し、初回の受信であるとされれば、処理ステップ S 84 へ移行し、受信されたリーフ更新情報 M s g. x 1' に示されるメッセージ I D が、例えば受信側レプリケータ 17 が有する、例えばメモリやハードディスクといった記録または記憶媒体に、コピー 8 として記憶される。そして、次のステップ S 85 で、受信されたリーフ更新情報 M s g. x 1' が受信側レプリケータ 17 での処理の対象として選択される。

【0149】

一方、ステップ S 82 で、上述のステップ S 80 でのリーフ更新情報 M s g. x 1' の受信が初回の受信では無いと判断されたら、処理はステップ S 83 に移行する。ステップ S 83 では、受信されたリーフ更新情報 M s g. x 1' のメッセージ I D が、前回までのリーフ更新情報 M s g. x 1' の受信によりステップ S 84 でコピー 8 として記憶媒体に記憶されたメッセージ I D と同一かどうか判断される。

【0150】

若し、両者が同一であると判断されたら、処理はステップ S 80 に戻される。一方、ステップ S 83 で両者が同一では無いと判断されたら、処理はステップ S 84 に移行し、今回受信されたコンテナ構造更新情報 M s g. x 1' のメッセージ I D が前回までのメッセージ I D の代わりに記憶媒体に記憶され、今回受信されたコンテナ構造更新情報 M s g. x 1' に基づき以降の処理が行われる。

【0151】

上述のようにして、受信側サーバ 16 で管理されるディレクトリ構造には、送信側サーバ 11 で管理されるディレクトリ構造のうち、受信側サーバ 16 を利用するユーザの嗜好を反映した部分のみを蓄積し、更新することができる。そのた

め、受信側サーバ 16 においてディレクトリ構造を蓄積するための蓄積記憶媒体を有効に利用することができる。また、それと共に、受信側サーバ 16 でのディレクトリ構造の格納コストを抑えることができる。さらに、受信側クライアント 15 による受信側サーバ 16 の内容の検索要求に対する処理効率を大幅に向上させることが可能となる。

【 0 1 5 2 】

なお、上述では、各コンテナエントリに割り当てられた割り当てマスク長を可変長としたが、これはこの例に限定されない。割り当てマスク長は、例えばバイト単位などの固定長とすることもできる。

【 0 1 5 3 】

次に、この発明の実施の一形態について説明する。この発明では、上述したディレクトリの放送型の差分更新方式を、実際の公開鍵証明書ディレクトリの放送型差分更新方式に適用する。まず、図 23 を用いて、公開鍵証明書ディレクトリについて説明する。図 23 は、ユーザ（加入者）と認証局とからなる階層構造である、認証局構造の一例を概略的に示す。

【 0 1 5 4 】

図 23 において、CA__0、CA__1 および CA__2 が認証局、EE__1、EE__2 および EE__3 がエンドエンティティ（加入者）であるとする。実線矢印は、矢印の元の認証局が矢印の先の認証局若しくはエンドエンティティの公開鍵の証明書を発行することを意味している。

【 0 1 5 5 】

図 23 の例では、認証局 CA__1 および CA__2 は、それぞれ認証局 CA__0 により認証され、公開鍵証明書を発行される。エンドエンティティ EE__1 および EE__2 は、認証局 CA__1 に認証され、公開鍵証明書を発行される。同様に、エンドエンティティ EE__3 は、認証局 CA__2 により認証され、公開鍵証明書を発行される。一方、認証局 CA__0 は、エンドエンティティ EE__1、EE__2 および EE__3 を直接的には認証していない。このように、各認証局およびエンドエンティティの間で階層構造が形成される。

【 0 1 5 6 】

また、点線矢印は、矢印の元のエンドエンティティが矢印の先の認証局と緊密な関係を結び、ルート認証局として信頼していることを意味する。すなわち、点線矢印の元のエンドエンティティは、矢印の先の認証局の公開鍵をルート公開鍵として利用する。図 2 3 の例では、例えば、エンドエンティティ E E _ 1 は、認証局 C A _ 0 と緊密な関係を結び、エンドエンティティ E E _ 1 は、ルート公開鍵として C A _ 0 の公開鍵を用いる。

【 0 1 5 7 】

ここで、一例として、エンドエンティティ E E _ 1 がエンドエンティティ E E _ 2 の公開鍵を得る場合について考える。この場合、エンドエンティティ E E _ 1 は、次の 2 つの証明書の証明書パスを処理することになる。すなわち、第 1 の証明書パスは、認証局 C A _ 0 によって発行された認証局 C A _ 1 の証明書を得るためのパスである。第 2 の証明書パスは、認証局 C A _ 1 によって発行されたエンドエンティティ E E _ 2 の証明書を得るためのパスである。

【 0 1 5 8 】

つまり、エンドエンティティ E E _ 1 にとっては、エンドエンティティ E E _ 2 の認証を行っている認証局 C A _ 1 が信頼できるかどうか分からないため、自分と信頼関係にある認証局 C A _ 0 に遡って認証局 C A _ 1 の認証を確認する必要がある。

【 0 1 5 9 】

また、他の例として、エンドエンティティ E E _ 1 がエンドエンティティ E E _ 3 の公開鍵を得る場合について考える。この場合、上述と同様に、エンドエンティティ E E _ 1 にとっては、エンドエンティティ E E _ 3 の認証を行っている認証局 C A _ 2 が信頼できるかどうか分からないため、エンドエンティティ E E _ 1 は、次の 2 つの証明書の証明書パスを処理することになる。すなわち、第 1 の証明書パスは、認証局 C A _ 0 によって発行された認証局 C A _ 2 の証明書を得るためのパスであり、第 2 のパスは、認証局 C A _ 2 によって発行されたエンドエンティティ E E _ 3 の証明書を得るためのパスである。

【 0 1 6 0 】

さらに他の例として、エンドエンティティ E E _ 2 がエンドエンティティ E E

__1 の公開鍵を得る場合について考える。この場合には、エンドエンティティ E E__1 および E E__2 は、共通の認証局 C A__1 によって認証されている。そのため、エンドエンティティ E E__2 は、認証局 C A__1 によって発行されたエンドエンティティ E E__1 の証明書を得るだけで事足りる。

【 0 1 6 1 】

図 2 3 から分かるように、認証局構造は、送信側ディレクトリサーバ 1 1 で管理されているディレクトリツリーに対応付けることが可能である。図 2 4 は、認証局構造とディレクトリツリーとの一例の対応関係を示す。図 2 4 A に示される認証局構造の各階層が、図 2 4 B に示されるディレクトリツリーの各階層にそれぞれ対応付けられる。すなわち、認証局構造における認証局 C A__0、C A__1、C A__2 がディレクトリツリーにおけるコンテナエントリ 1 0 0、1 0 1 および 1 0 2 にそれぞれ対応付けられる。認証局構造におけるエンドエンティティ E E__1、E E__2 および E E__3 がリーフエントリ 1 1 0、1 1 1 および 1 1 2 にそれぞれ対応付けられる。

【 0 1 6 2 】

ディレクトリツリーの各エントリに対するエントリ名も、認証局構造に対応して与えることができる。例えば、図 3 を用いて上述した命名法を倣えば、コンテナエントリ 1 0 0 のエントリ名は、対応する認証局 C A__0 に基づきエントリ名 C A__0 とされる。コンテナエントリ 1 0 1 は、認証局構造において認証局 C A__0 から階層構造を辿った認証局 C A__1 に対応している。したがって、コンテナエントリ 1 0 1 には、エントリ名 C A__0、C A__1 が与えられる。同様に、リーフエントリ 1 1 0 は、認証局構造において、認証局 C A__1 からさらに階層構造を辿ったエンドエンティティ E E__1 に対応している。したがって、リーフエントリ 1 1 0 には、エントリ名 C A__0、C A__1、E E__1 が与えられる。他のエントリにも、同様にしてに消極構造に対応してエントリ名を与えることができる。

【 0 1 6 3 】

認証局構造における認証局に対応するエントリ（以下、C A エントリと称する）には、以下の 2 つの属性を持たせる。1 つは、対応する認証局の最新の公開鍵

証明書のシリアル番号を格納する属性である。ここでは、この属性の属性名を"PublicKeyCertificateSerialNumber"とし、属性値をシリアル番号そのものとする。

【 0 1 6 4 】

もう1つは、対応する認証局の最新の公開鍵証明書、若しくは、最新の公開鍵証明書を取得する取得方法を格納する属性である。ここでは、この属性の属性名を"LatestPublicKeyCertificate"とし、属性値を、最新の公開鍵証明書そのもの、若しくは、最新の公開鍵証明書の取得方法とする。最新の公開鍵証明書を取得する取得方法は、例えば、その公開鍵証明書が置かれているURLで示される。

【 0 1 6 5 】

一方、認証局構造におけるエンドエンティティに対応するエントリ（以下、Eエントリと称する）には、同様にして、以下の2つの属性を持たせる。1つは、対応するエンドエンティティの最新の公開鍵証明書のシリアル番号を格納する属性である。ここでは、この属性の属性名を"PublicKeyCertificateSerialNumber"とし、属性値を、シリアル番号そのものとする。

【 0 1 6 6 】

もう1つは、対応するエンドエンティティの最新の公開鍵証明書、若しくは、最新の公開鍵証明書の取得方法を格納する属性である。ここでは、この属性の属性名を"LatestPublicKeyCertificate"とし、属性値を、最新の公開鍵証明書そのもの、若しくは、最新の公開鍵証明書の取得方法とする。最新の公開鍵証明書を取得する取得方法は、例えば、その公開鍵証明書が置かれているURLで示される。

【 0 1 6 7 】

上述のように、この実施の一形態では、エントリの属性に対して、最新の公開鍵証明書を直接的に格納する場合と、最新の公開鍵証明書を取得するための情報を格納する場合との、2種類の形態が設けられている。このように、エントリ属性に対して2種類の情報を格納可能としておけば、放送ネットワーク資源を有効的に活用できる。

【 0 1 6 8 】

一般に、公開鍵証明書は、その証明書を取得するための情報（例えばURLなど）に比べてデータサイズが非常に大きい。したがって、ディレクトリ構造が極めて多数のディレクトリエントリを有する場合には、それら多数のディレクトリエントリに対応する最新の公開鍵証明書を周期的に放送すると、帯域を大幅に圧迫してしまうことになる。しかしながら、帯域を節約するために、全て公開鍵証明書の取得情報のみの放送としてしまうと、受信側における最新の公開鍵の参照時に、通信ネットワークを介した公開鍵証明書の取得処理が必ず行われることになり、通信ネットワークのトラフィックを圧迫してしまう。

【 0 1 6 9 】

そこで、放送ネットワークを介した即時広域配布のメリットを生かすために、この発明の実施の一形態では、以下のような方法を採用。すなわち、例えばとある公開鍵証明書が失効した場合、新たに発行された公開鍵証明書を、対応するエントリの属性"LatestPublicKeyCertificate"に格納すると共に、その公開鍵証明書のシリアル番号を"PublicKeyCertificateSerialNumber"に格納し、更新情報として放送する。所定の時間が経過したら、当該公開鍵証明書をとあるURLで指定されるアドレスに置き、エントリの属性"LatestPublicKeyCertificate"の内容を当該URLに入れ替え、更新情報として放送する。こうすることで、帯域を有効活用することが可能となる。

【 0 1 7 0 】

図 2 5 は、この実施の一形態に適用可能な系の一例を示す。この図 2 5 の例は、暗号化電子メールなどの暗号化通信に 2 つのエンドエンティティが関わる例である。また、この図では、認証局CA__0およびCA__1が関わる。なお、図 2 5 において、上述した図 1 と共通する部分には同一の番号を付し、詳細な説明を省略する。

【 0 1 7 1 】

図 2 5 において、受信側 3' をなす受信側レプリケータ 1 7'、受信側サーバ 1 6' および受信側クライアント 1 5' は、受信側 3 をなす上述した受信側レプリケータ 1 7、受信側サーバ 1 6 および受信側クライアント 1 5 と同等なもので、放送ネットワーク 2 を介して送信側 1 の送信側サーバ 1 1 で管理されるディレ

クトリツリーの複製を管理する。

【 0 1 7 2 】

ここで、受信側クライアント 1 5 および 1 5' が図 2 3 および図 2 4 で上述したエンドエンティティ E E _ 1 および E E _ 2 にそれぞれ対応するものとする。受信側クライアント 1 5 および 1 5' は、それぞれ、例えばソフトウェアからなる暗号化通信モジュール 5 0 および 5 0' を有する。受信側クライアント 1 5 と 1 5' は、暗号化通信モジュール 5 0 および 5 0' を用いて、通信ネットワーク 5 1 を介して互いに暗号化通信を行う。

【 0 1 7 3 】

さらに、送信側において、送信側クライアント 1 0 および 1 0' が存在し、送信側クライアント 1 0' は、送信側クライアント 1 0 と同様に、送信側サーバ 1 1 で管理されるディレクトリツリーの内容を更新できる。これら送信側クライアント 1 0 および 1 0' は、図 2 3 および図 2 4 で上述した、認証局階層における認証局 C A _ 0 および C A _ 1 に対応する。

【 0 1 7 4 】

なお、図 2 5 では、受信側は、受信側 3 および 3' の 2 つが示されているが、実際には、さらに多数の受信側構成が存在し、これら多数の受信側構成は、通信ネットワーク 5 1 を介して互いに通信が可能とされている。また、送信側クライアントも、ここでは送信側クライアント 1 0 および 1 0' の 2 つが示されているが、実際には、さらに多数が存在する。

【 0 1 7 5 】

図 2 6 は、受信側クライアント 1 5 および 1 5' の間で行われる暗号化通信の一例の手順を示すフローチャートである。なお、図 2 6 では、受信側レプリケータ 1 7 および 1 7' をそれぞれ第 1 および第 2 の受信側レプリケータ、暗号化通信モジュール 5 0 および 5 0' をそれぞれ第 1 および第 2 の暗号化通信モジュールと称している。

【 0 1 7 6 】

まず、このフローチャートの処理に先立ち、受信側クライアント 1 5 および 1 5' は、それぞれ、自身の秘密鍵および公開鍵のペアを生成しているものとする

。また、受信側クライアント 1 5 すなわちエンドエンティティ E E _ 1 は、送信側クライアント 1 0' すなわち認証局 C A _ 1 から、自身の公開鍵について公開鍵証明書を発行してもらっているものとする。このとき、図 2 3 を用いて上述した証明書パス、すなわち、認証局 C A _ 0 が認証局 C A _ 1 に証明書を発行し、認証局 C A _ 1 がエンドエンティティ E E _ 2 の証明書を発行することを考慮に入れ、前提として、認証局 C A _ 1 は、認証局 C A _ 0 すなわち送信側クライアント 1 0 から予め公開鍵証明書を発行してもらっているものとする。

【 0 1 7 7 】

同様に、受信側クライアント 1 5' すなわちエンドエンティティ E E _ 2 は、送信側クライアント 1 0' すなわち認証局 C A _ 1 から、自身の公開鍵について公開鍵証明書を発行してもらっているものとする。

【 0 1 7 8 】

これらの、受信側クライアント 1 5 および 1 5' 各々の公開鍵についての公開鍵証明書のそれぞれは、受信側クライアント 1 5 および 1 5' が共にアクセス可能な場所、例えばインターネット上のある W e b サイトに格納するか、予め相互に交換し合うことにより、互いに必要なときに利用できるようにしておく。

【 0 1 7 9 】

まず、最初のステップ S 9 0 で、受信側クライアント 1 5 において、暗号通信モジュール 5 0 が利用され受信側クライアント 1 5 の秘密鍵によってメッセージに署名される。次のステップ S 9 1 では、受信側クライアント 1 5 により受信側クライアント 1 5' の公開鍵証明書が取得され、取得された公開鍵証明書に格納されている公開鍵が得られる。なお、ステップ S 9 1 の処理の前提として、受信側クライアント 1 5 により送信側クライアント 1 0' の公開鍵証明書が取得され、そこに格納されている公開鍵が既に取得されている。次のステップ S 9 2 では、受信側クライアント 1 5 において、暗号通信モジュール 5 0 が用いられ、ステップ S 9 1 で得られた公開鍵によりステップ S 9 0 で署名されたメッセージが暗号化される。

【 0 1 8 0 】

ステップ S 9 2 で暗号化されたメッセージは、ステップ S 9 3 で、通信ネット

ワーク 5 1 を介して受信側クライアント 1 5' に送信される。受信側クライアント 1 5' では、受信側クライアント 1 5 から送信されたメッセージを受信し、暗号通信モジュール 5 0' を用いて受信側クライアント 1 5' の秘密鍵で、受信されたメッセージを復号化する。復号化されたメッセージには、上述のステップ S 9 0 において、受信側クライアント 1 5 の秘密鍵を用いて署名されている。

【 0 1 8 1 】

次のステップ S 9 5 では、受信側クライアント 1 5' により、受信側クライアント 1 5 の公開鍵証明書が取得され、取得された公開鍵証明書に格納されている公開鍵が得られる。そして、ステップ S 9 6 で、受信側クライアント 1 5' により、暗号化モジュール 5 0' が用いられ、ステップ S 9 5 で得られた公開鍵により上述のステップ S 9 4 で復号化されたメッセージの署名が確認される。

【 0 1 8 2 】

上述したフローチャートのうち、ステップ S 9 1 およびステップ S 9 5 において公開鍵が取得される際に、受信側クライアント 1 5 および 1 5' は、取得する公開鍵が失効していない有効な公開鍵証明書に基づくものであることを確認しなければならない。この確認は、これらの公開鍵証明書を発行した認証局に問い合わせることによって可能である。しかしながら、一般的には、公開鍵証明書の発行数が多い認証局では、問い合わせの負荷が増大し、応答性能が極端に低下するという問題により、オンラインによる有効性の確認処理が不可能であることが多い。

【 0 1 8 3 】

そこで、この発明の実施の一形態では、受信側クライアント 1 5 や受信側クライアント 1 5' は、それぞれ利用する公開鍵の有効性について該当する認証局に問い合わせる代わりに、各自のローカルな環境に設置されている受信側サーバ 1 6 や受信側サーバ 1 6' で管理されるディレクトリに問い合わせるようにする。受信側クライアント 1 5 や 1 5' のローカルな環境に設置されている受信側サーバ 1 6 や 1 6' は、例えば、家庭内 LAN (Local Area Network) 上、マンションなどに設置される電話線集線装置、ケーブルテレビネットワークのヘッドエンドなどが想定できる。これにより、当該公開鍵証明書を発行した認証局への問い合わせの集中を分散化させることができる。

【 0 1 8 4 】

上述した、認証局への問い合わせをローカルな環境に置き換えて行うには、以下の情報を更新する必要がある。第 1 に、受信側サーバ 1 6 においては、受信側クライアント 1 5' すなわちエンドエンティティ EE__2 の公開鍵証明書が有効か否かを示す情報が更新されている必要がある。また、この例では、上述した証明書パスに従い、受信側クライアント 1 5 すなわちエンドエンティティ EE__1 は、エンドエンティティ EE__2 に公開鍵証明書を発行する認証局 CA__1 とは緊密な関係にない。したがって、受信側サーバ 1 6 においては、送信側クライアント 1 0' 、すなわち、エンドエンティティ EE__2 に対して公開鍵証明書を発行する認証局 CA__1 の公開鍵証明書が有効であるか否かを示す情報も、最新の情報に更新されている必要がある。

【 0 1 8 5 】

第 2 に、受信側サーバ 1 6' においては、受信側クライアント 1 5 すなわちエンドエンティティ EE__1 の公開鍵証明書が有効であるか否かの情報が、最新の情報に更新されている必要がある。

【 0 1 8 6 】

なお、公開鍵証明書が有効であるかどうか、すなわち、その公開鍵証明書が失効しているか否かは、当該公開鍵証明書と、当該公開鍵証明書に対応する最新の公開鍵証明書とを比較し、シリアル番号が一致しているかどうかを調べることで判断できる。この発明では、新たに発行された公開鍵証明書のシリアル番号を、エントリの属性 "PublicKeyCertificateSerialNumber" に格納するようにしているため、このエントリ属性を調べることで、当該公開鍵証明書が有効であるかどうかを知ることができる。

【 0 1 8 7 】

上述の図 2 4 に示されるように、図 2 3 に示される認証局構造と、受信側クライアント 1 5 および 1 5' のディレクトリエントリとの対応付けがなされている。したがって、受信側サーバ 1 6 で管理されるディレクトリにおいては、コンテナエントリ 1 0 1 (エントリ名 CA__0, CA__1) およびリーフエントリ 1 1 1 (エントリ名 CA__0, CA__1, EE__2) の更新情報に注目する必要がある。

る。同様に、受信側サーバ 1 6' で管理されるディレクトリにおいては、リーフエントリ 1 1 0 の更新情報を注目しなければならない。

【 0 1 8 8 】

受信側サーバ 1 5 および 1 5' のそれぞれでは、上述の注目対象となるエントリの上位のコンテナエントリがフィルタリングマスクの対象コンテナエントリとして選択される。フィルタリングマスクの対象コンテナエントリの選択は、上述した図 2 1 に従い、ステップ S 7 4 においてなされる。選択されたコンテナエントリに対応するフィルタリングマスクが、それぞれのターゲットマスクリストに格納されることになる。

【 0 1 8 9 】

図 2 7 を用いて、それぞれのターゲットマスクリストにどのようなフィルタリングマスクが格納されるかについて、より具体的に説明する。なお、図 2 7 において、斜線が付された四角は、公開鍵証明書を取得する対象となるエントリを示す。また、点線で示された四角は、フィルタリングマスクに対応するコンテナエントリを示す。図 2 7 A および図 2 7 B は、それぞれ受信側サーバ 1 6 および 1 6' におけるフィルタリング対象のエントリを示す。

【 0 1 9 0 】

受信側サーバ 1 6 におけるフィルタリング対象のエントリは、図 2 7 A に示されるように、エントリ名 CA__0、CA__1 のコンテナエントリ 1 0 1 と、エントリ名 CA__0、CA__1、EE__2 のリーフエントリ 1 1 1 である。受信側レプリケータ 1 7 によって、これらエントリ 1 0 1 および 1 1 1 をそれぞれ示すフィルタリングマスクがターゲットマスクリストに格納される。ターゲットマスクリストは、例えば受信側レプリケータ 1 7 が有する記憶媒体に記憶される。

【 0 1 9 1 】

同様にして、受信側サーバ 1 6' におけるフィルタリングフィルタリング対象のエントリは、図 2 7 B に示されるように、エントリ名 CA__0、CA__1、EE__1 のリーフエントリ 1 1 0 である。受信側レプリケータ 1 7' によって、これらエントリ 1 1 0 を示すフィルタリングマスクがターゲットマスクリストに格納される。

【 0 1 9 2 】

なお、送信側サーバ 1 1 で管理されているディレクトリ情報と、受信側サーバ 1 6 および 1 6' で管理されているディレクトリ情報とは、同期がとれている必要がある。この実施の一形態では、既に述べたように、所定の時間にセットされたタイマによって決められるタイミングで以て放送される、ディレクトリ情報の差分更新情報によって受信側サーバ 1 6 および 1 6' で管理されているディレクトリ情報が更新されることで、送信側と受信側との同期が取られる。

【 0 1 9 3 】

このとき、送信側と受信側の同期は、段階的にとるようにできる。例えば、上述のタイミングで差分更新情報による更新を行うと共に、より長い周期で全てのディレクトリ構造の放送を行い、受信側サーバ 1 6 および 1 6' の更新を行うようにできる。さらに、受信側サーバ 1 6 および 1 6' と送信側サーバ 1 1 とを双方向通信が可能な通信回線で接続して、受信側サーバ 1 6 および 1 6' 側において、ターゲットマスキリストに格納されている情報については、随時、送信側サーバ 1 1 に対して最新情報を要求し、送信側サーバ 1 1 では、この要求に応じて、対応するディレクトリ構造の送信を行うようにできる。

【 0 1 9 4 】

【発明の効果】

以上説明したように、この発明の実施の一形態では、ターゲットマスキリストを用いることによって、上述のようにして、受信側サーバ 1 6 および 1 6' で管理されるディレクトリツリーの内容には、送信側サーバ 1 1 で管理されるディレクトリツリー構成のうち、受信側クライアント 1 5 および 1 5' がよくアクセスするエントリの情報だけを蓄積および更新することができる。これにより、受信側サーバ 1 6 および 1 6' におけるディレクトリツリーの格納コストを抑えることができると共に、蓄積媒体の有効活用が図れる効果がある。

【 0 1 9 5 】

また、この発明によれば、認証局およびエンドエンティティからなる階層構造である認証局構造と、送信側ディレクトリサーバおよび受信側ディレクトリサーバで管理されるディレクトリツリーとが対応付けられ、認証局構造において公開

鍵証明書を得るための証明書パスに対応してターゲットマスキリストが作成される。そのため、複数の受信側ディレクトリサーバの内容に対する複数の受信側ディレクトリサーバからの公開鍵証明書の失効情報検索用急に対する処理効率を、大幅に向上させることができる効果がある。

【図面の簡単な説明】

【図 1】

この発明に適用できる系の一例を示す略線図である。

【図 2】

複数の受信側が放送ネットワークに接続されることを説明するための略線図である。

【図 3】

ディレクトリ構造を説明するための略線図である。

【図 4】

コンテナエントリの構造の一例を示す略線図である。

【図 5】

リーフエントリの構造の一例を示す略線図である。

【図 6】

送信側レプリケータの機能を説明するための機能ブロック図である。

【図 7】

受信側クライアントの機能を説明するための機能ブロック図である。

【図 8】

受信側サーバの機能を説明するための機能ブロック図である。

【図 9】

ディレクトリ構造の差分更新情報について説明するための略線図である。

【図 10】

ディレクトリ構造の差分更新情報について説明するための略線図である。

【図 11】

コンテナエントリの同期管理方法を説明するためのフローチャートである。

【図 12】

コンテナエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図 1 3】

コンテナエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図 1 4】

リーフエントリの同期管理方法を説明するためのフローチャートである。

【図 1 5】

リーフエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図 1 6】

リーフエントリの同期管理方法をより詳細に説明するためのフローチャートである。

【図 1 7】

フィルタリングマスクのマスク値のビット配列構造を説明するための略線図である。

【図 1 8】

エントリの追加や削除が行われた場合の、コンテナエントリの増減に応じたマスク値の割り当て処理のフローチャートである。

【図 1 9】

コンテナエントリマスクスキーマの符号化を説明するための略線図である。

【図 2 0】

ターゲットマスクリストを説明するための略線図である。

【図 2 1】

ターゲットマスクリストを作成する処理のフローチャートである。

【図 2 2】

ターゲットマスクリストに基づき、放送されたリーフ更新情報 $Msg. \times 1'$ を選択的に受信する処理を示すフローチャートである。

【図 2 3】

認証局構造を説明するための図である。

【図 2 4】

認証局構造とディレクトリツリーとの一例の対応関係を示す略線図である。

【図 2 5】

実施の一形態に適用可能な系の一例を示す略線図である。

【図 2 6】

受信側クライアントの間で行われる暗号化通信の一例の手順を示すフローチャートである。

【図 2 7】

それぞれのターゲットマスキリストにどのようなフィルタリングマスクが格納されるかについて説明するための図である。

【図 2 8】

公開鍵証明書仕組みについて説明するための図である。

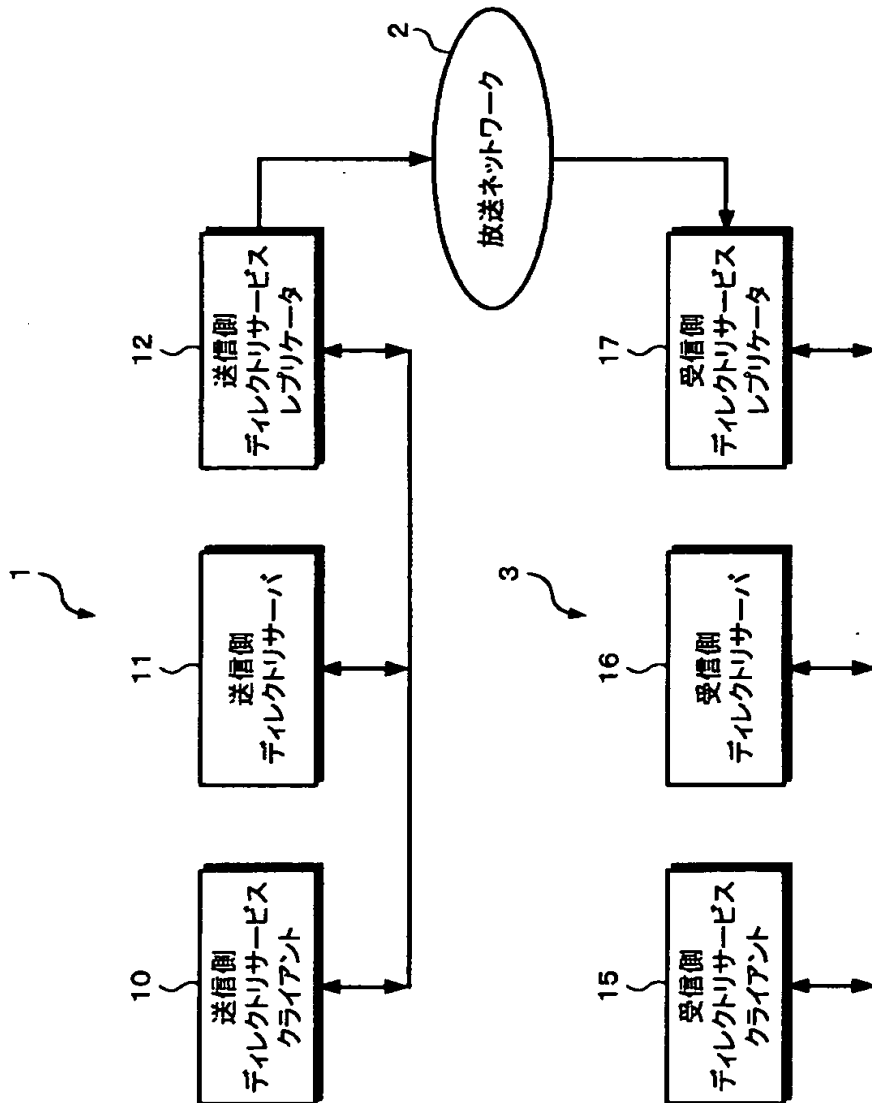
【符号の説明】

1・・・送信側、2・・・放送ネットワーク、3・・・受信側、10, 10'・・・送信側ディレクトリサービスクライアント、11・・・送信側ディレクトリサーバ、12・・・送信側ディレクトリサーバレプリケータ、15, 15'・・・受信側ディレクトリサービスクライアント、16, 16'・・・受信側ディレクトリサーバ、17, 17'・・・受信側ディレクトリサーバレプリケータ、50, 50'・・・暗号化通信モジュール、51・・・通信ネットワーク、CA__0, CA__1, CA__2・・・認証局、EE__1, EE__2, EE__3・・・エンドエンティティ

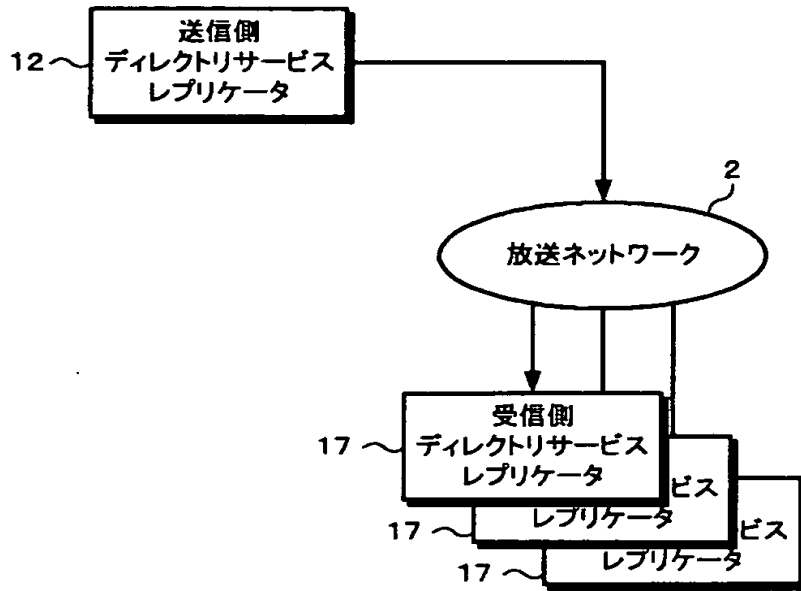
【書類名】

図面

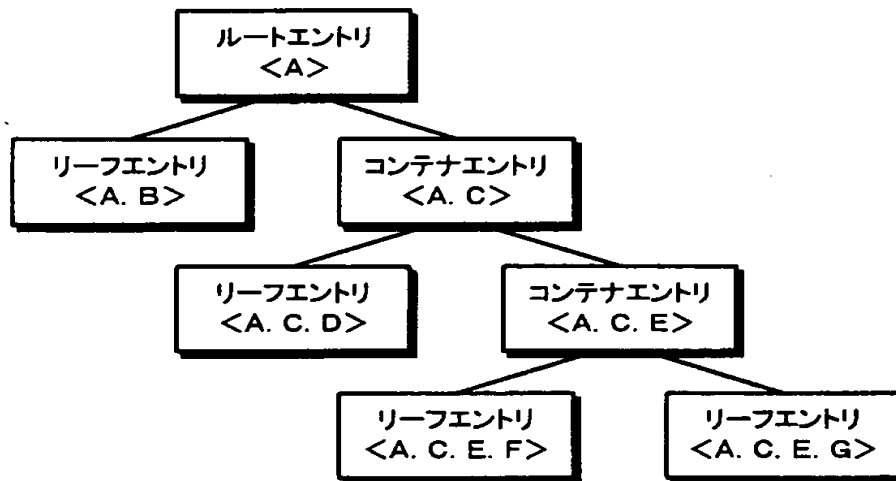
【図 1】



【図 2】

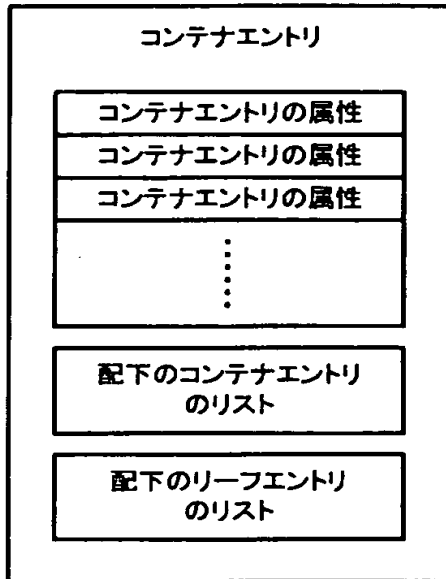


【図 3】

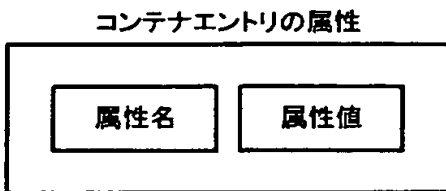


【図 4】

A

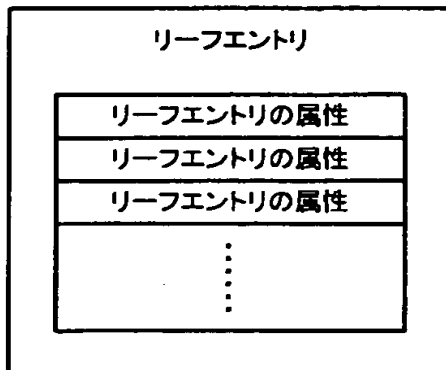


B

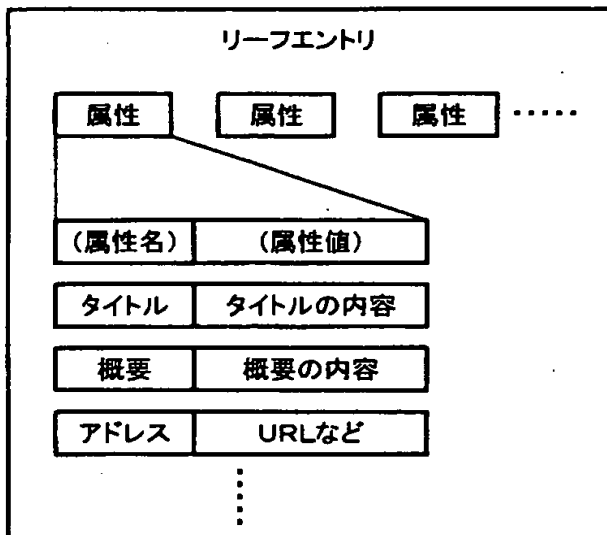


【図 5】

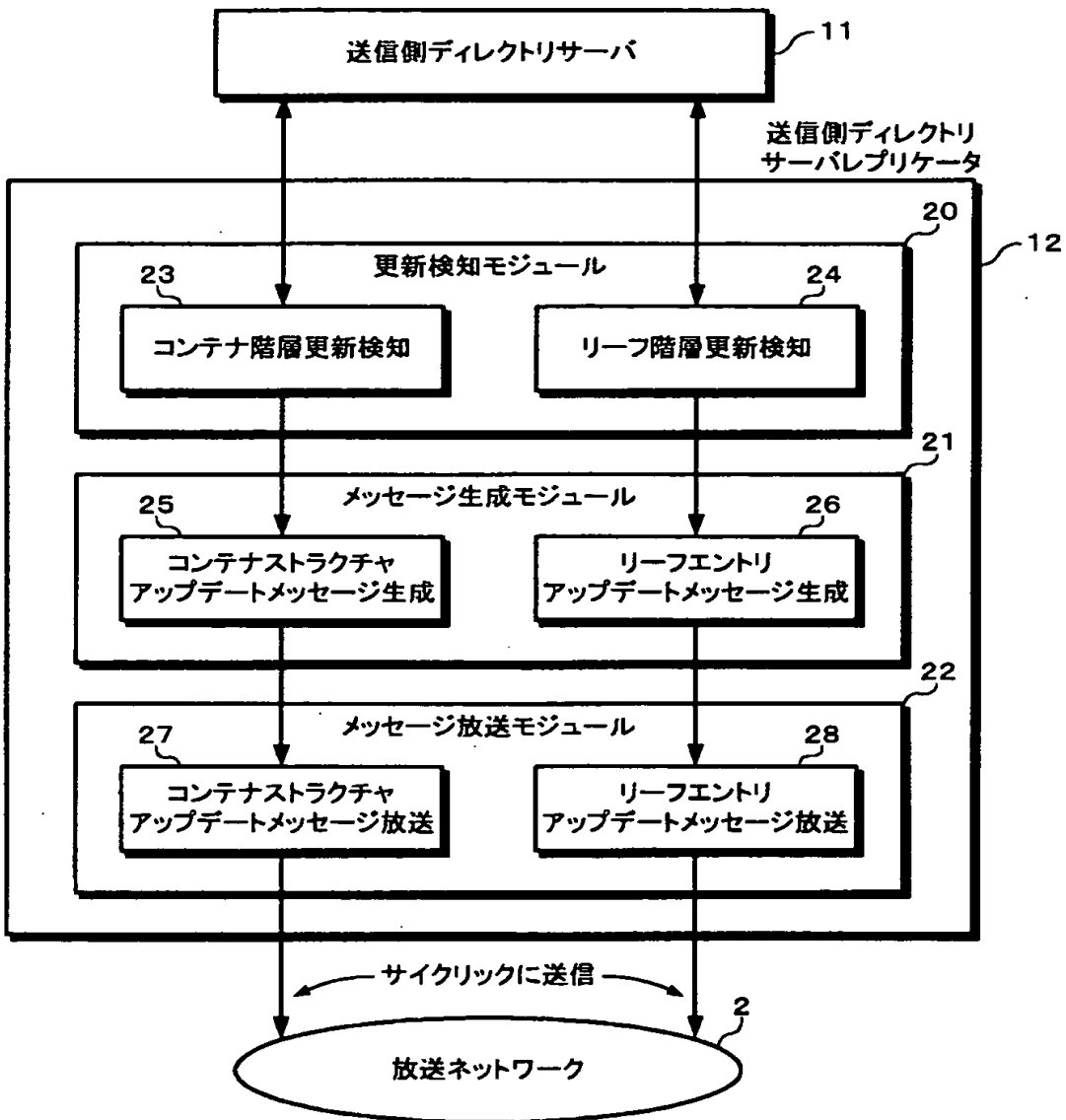
A



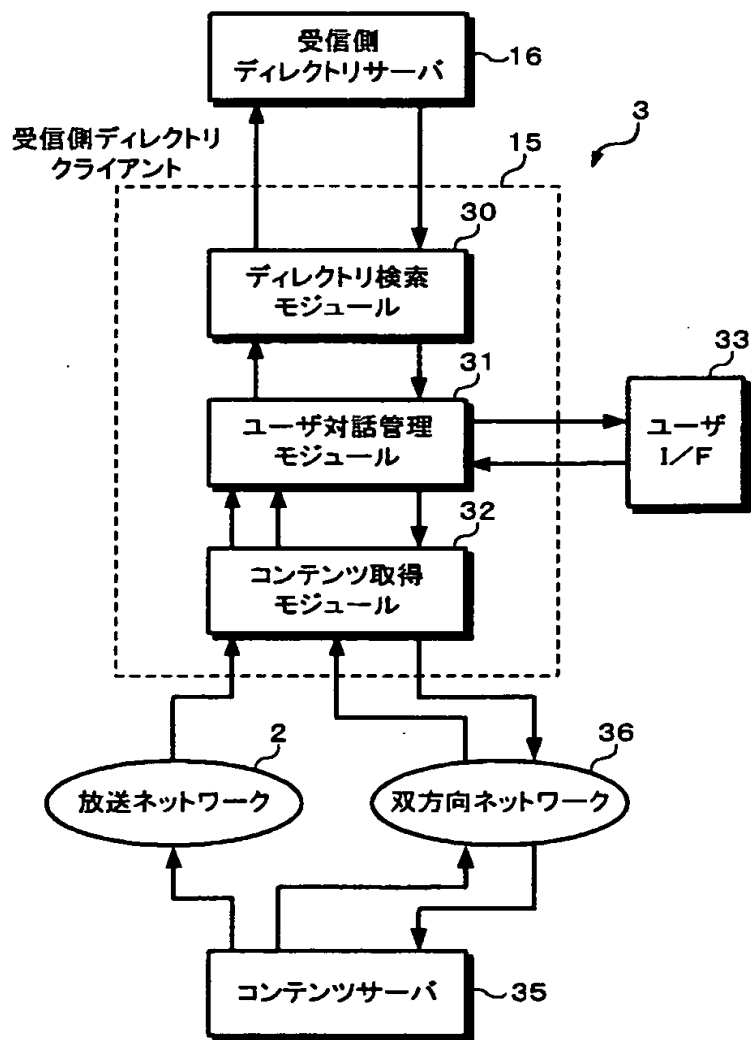
B



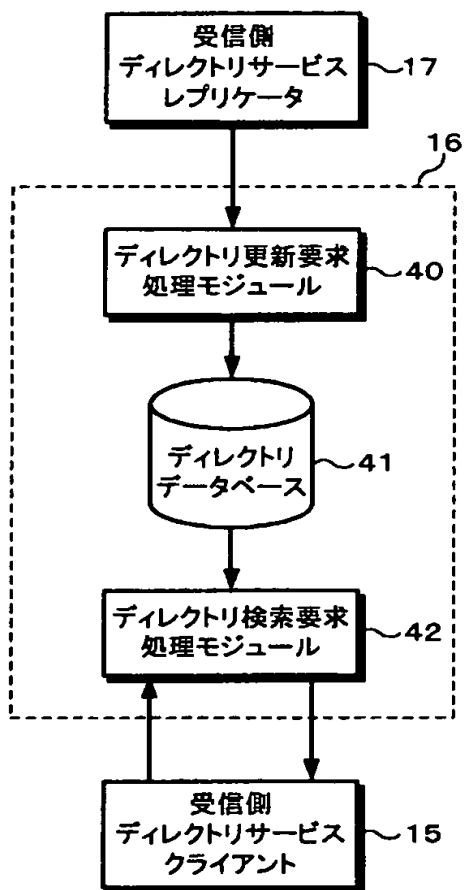
【図 6】



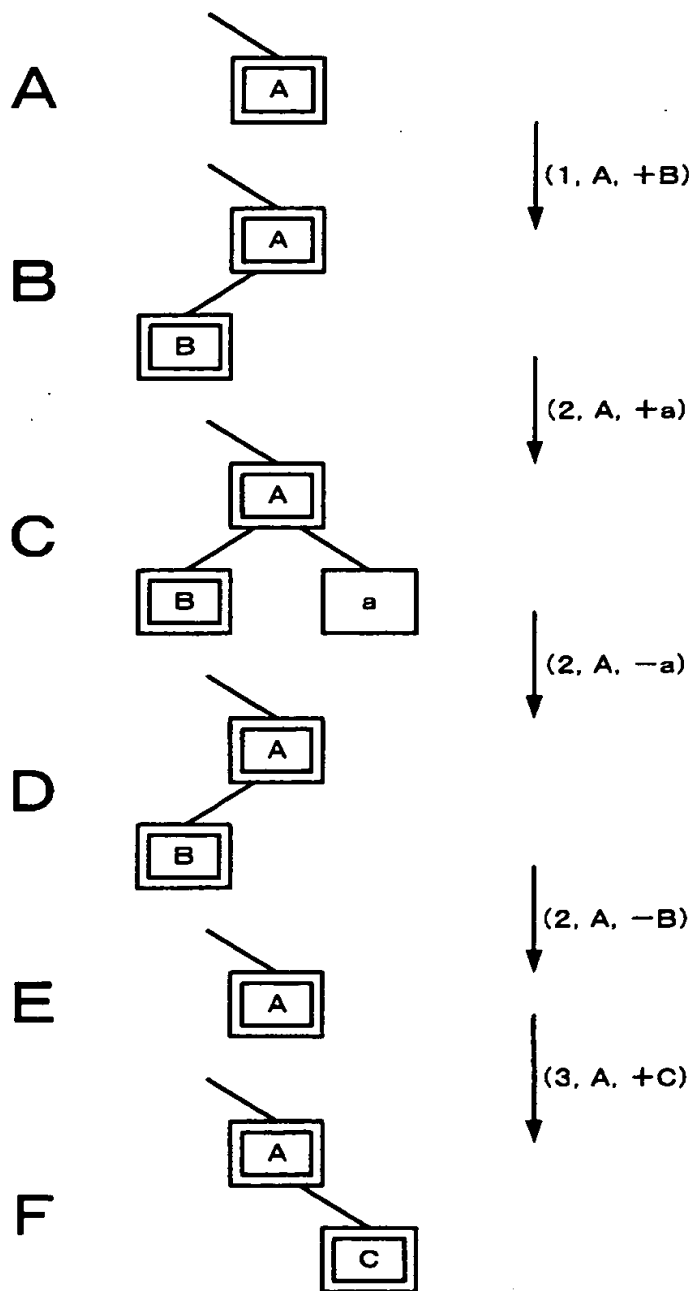
【図 7】



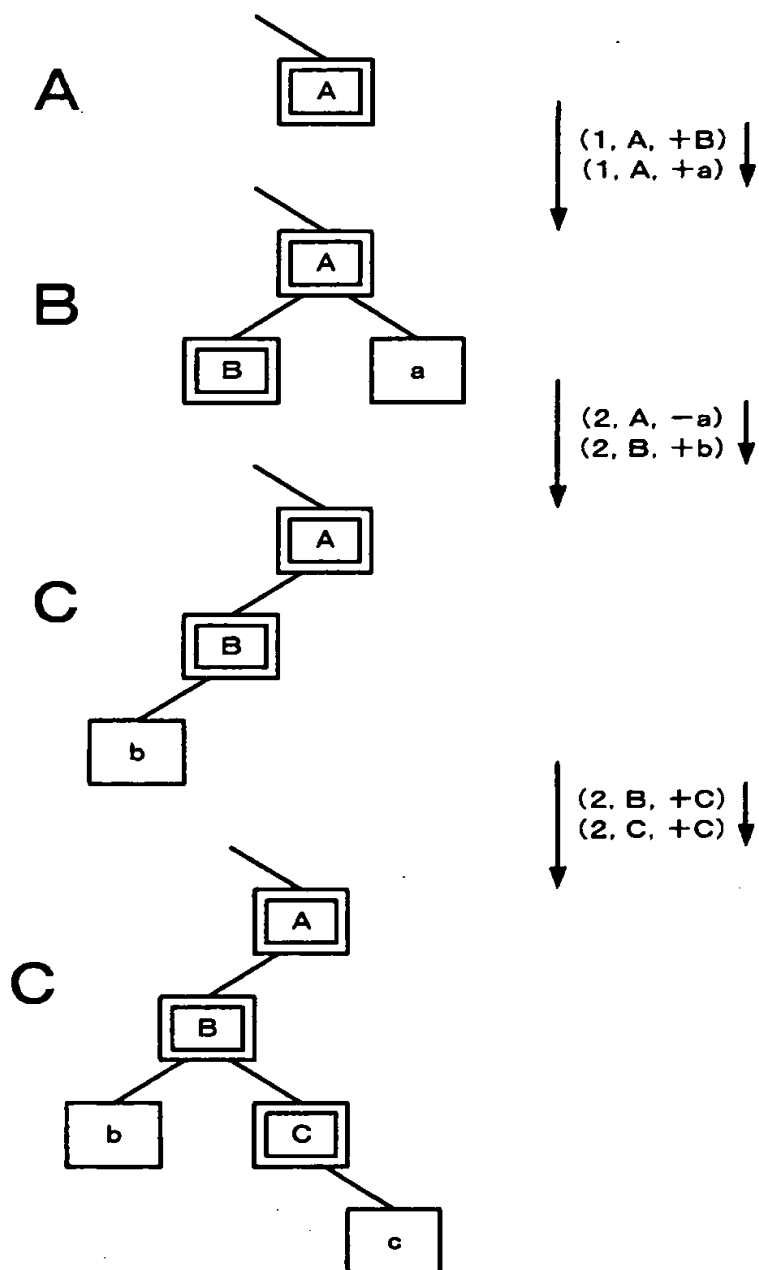
【図 8】



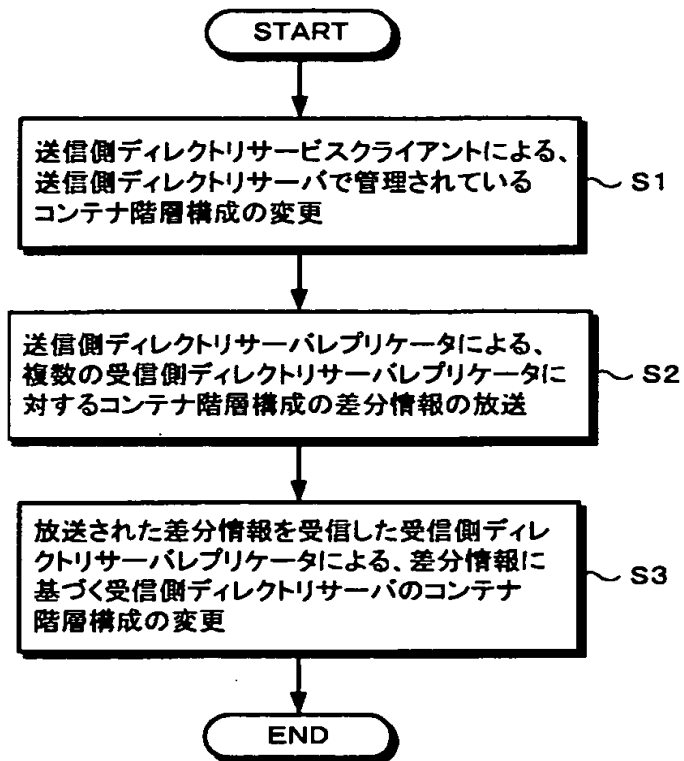
【図 9】



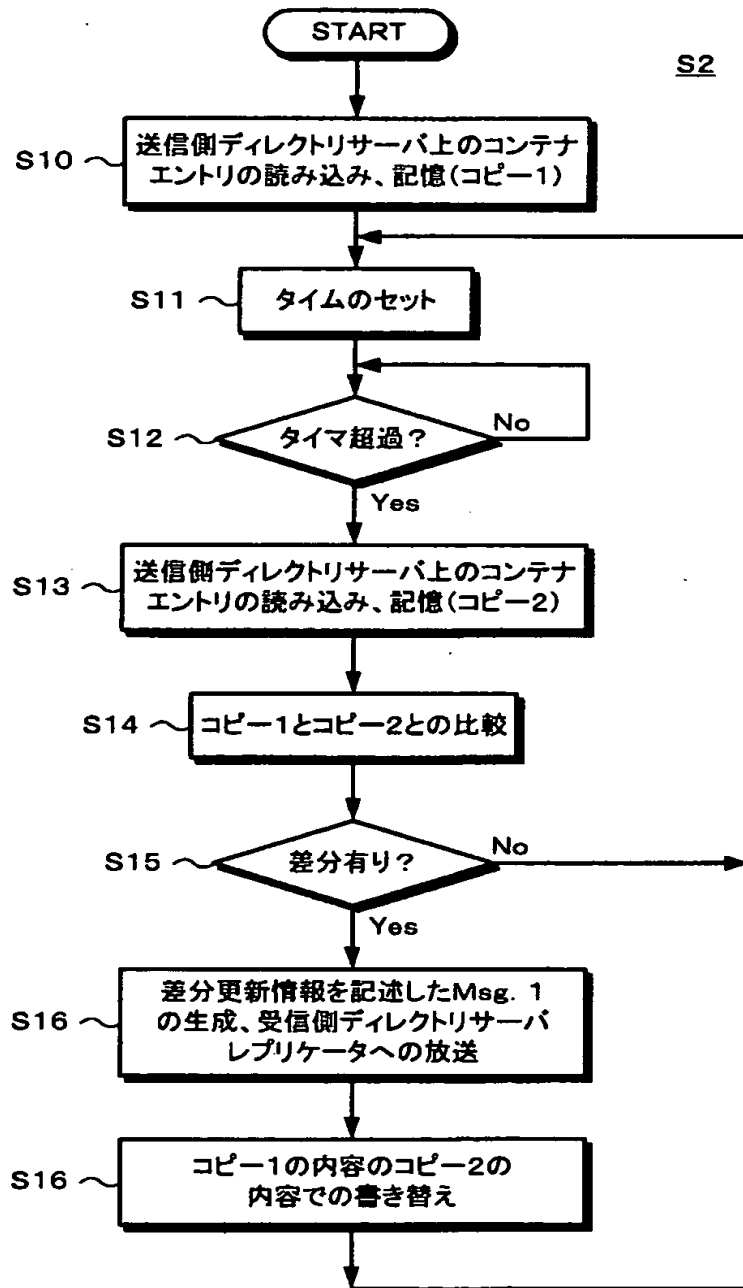
【図10】



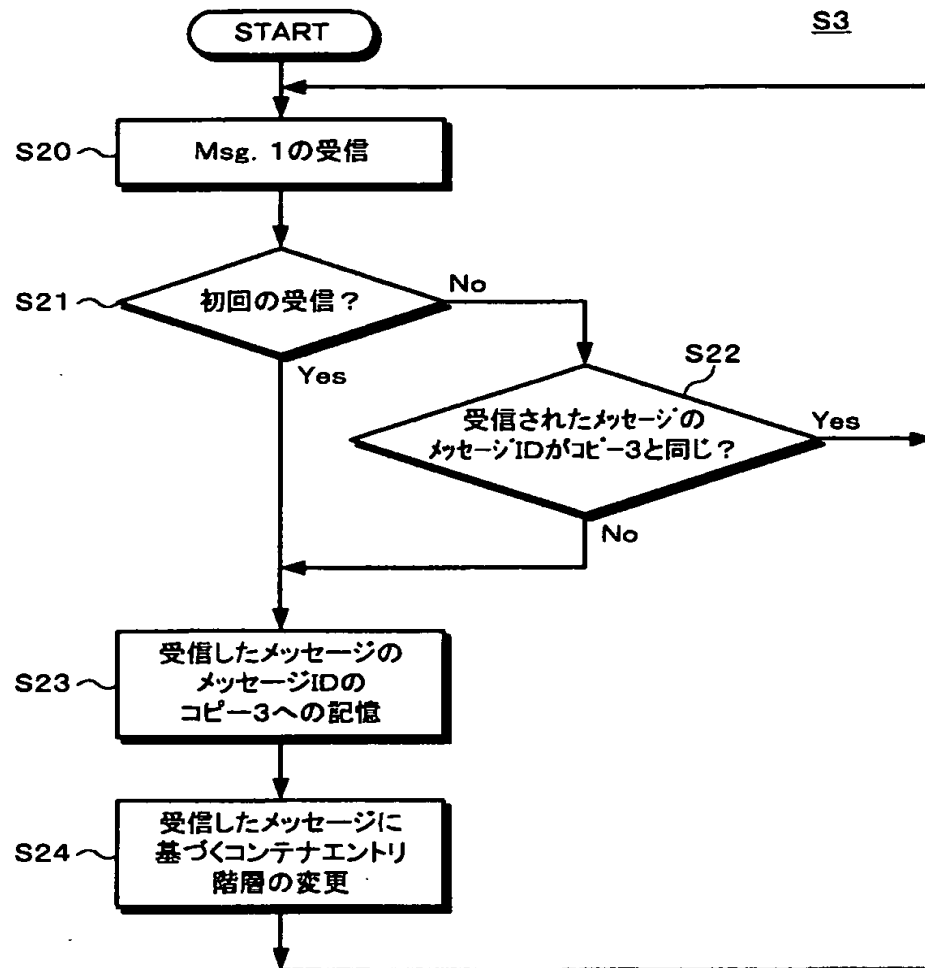
【図 1 1】



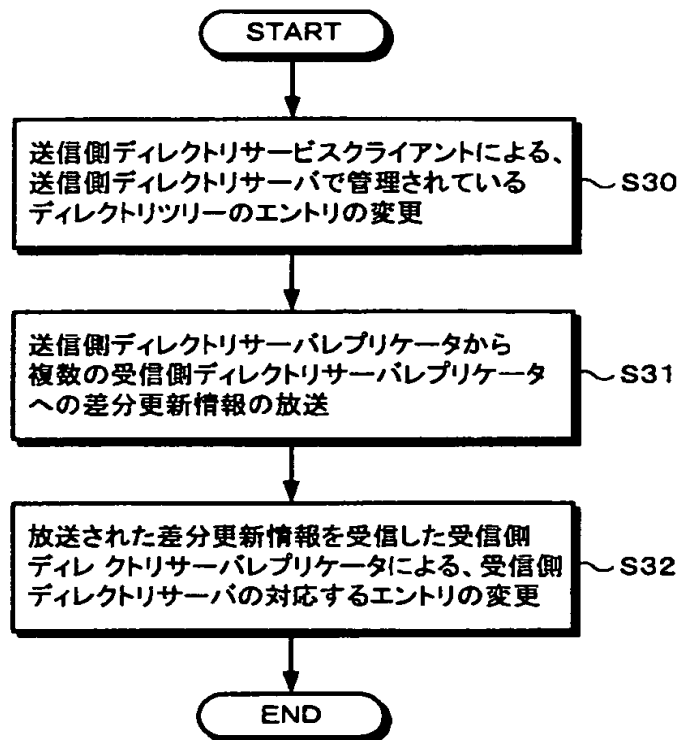
【図 12】



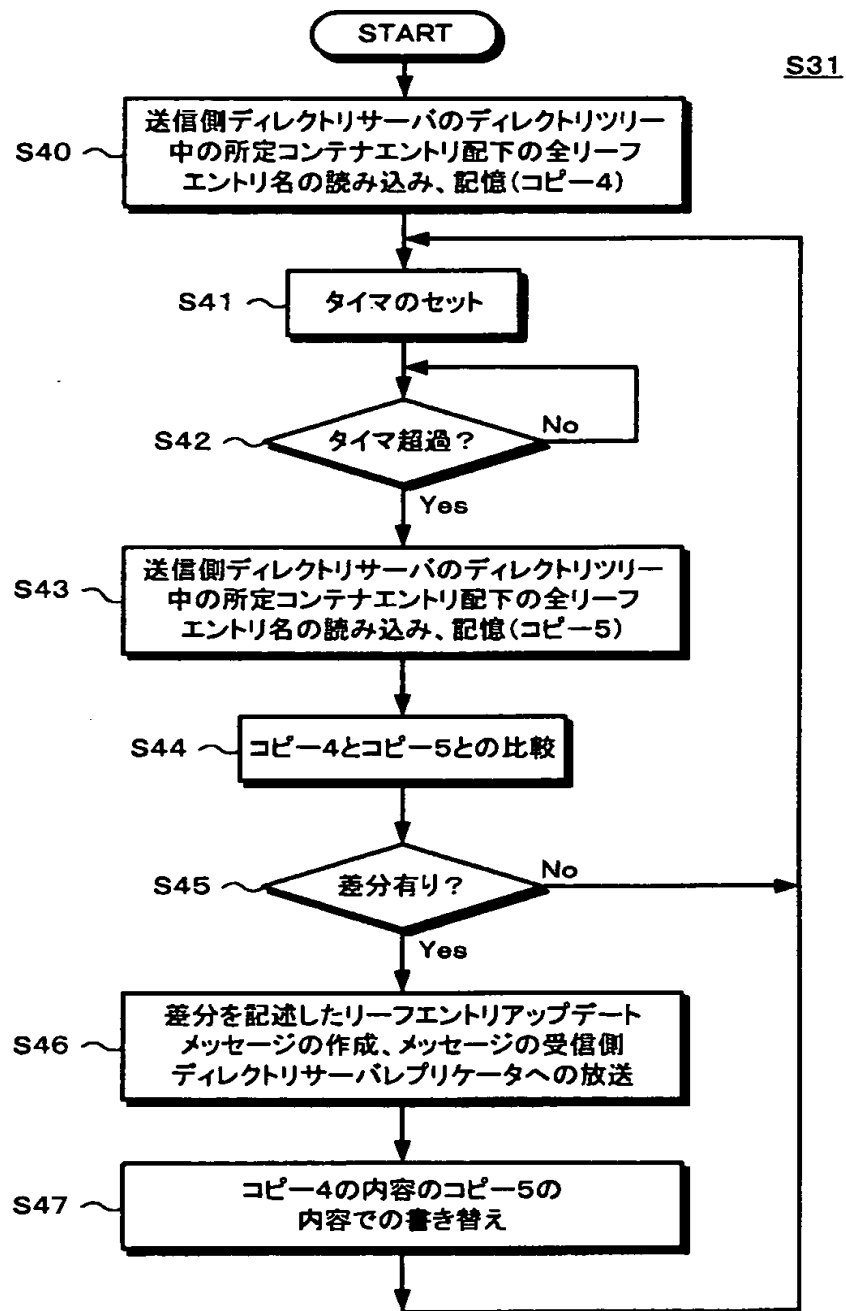
【図 1 3】



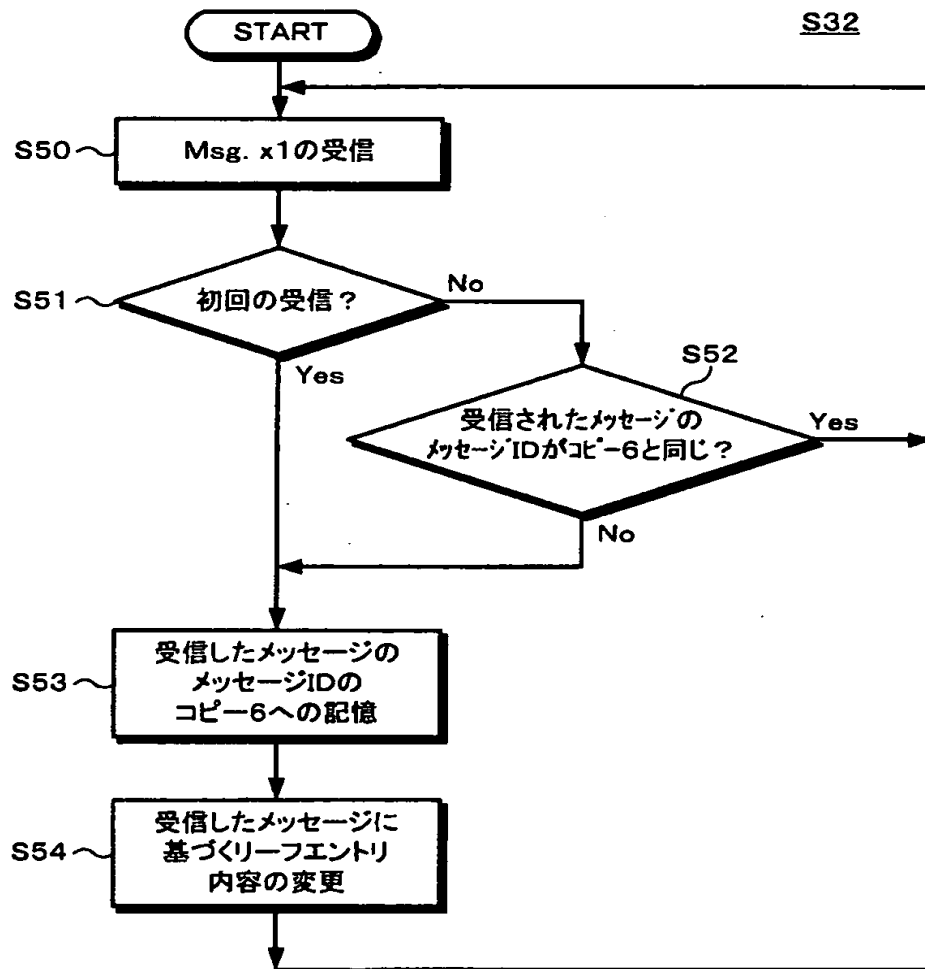
【図 1 4】



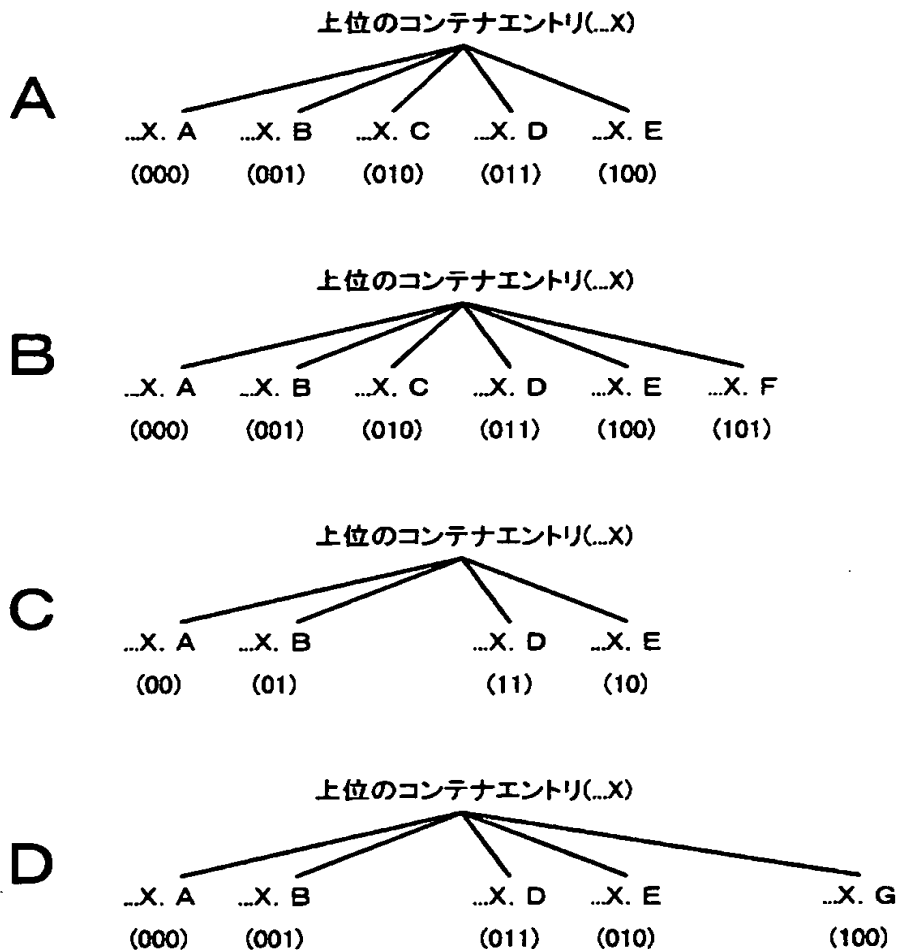
【図 1 5】



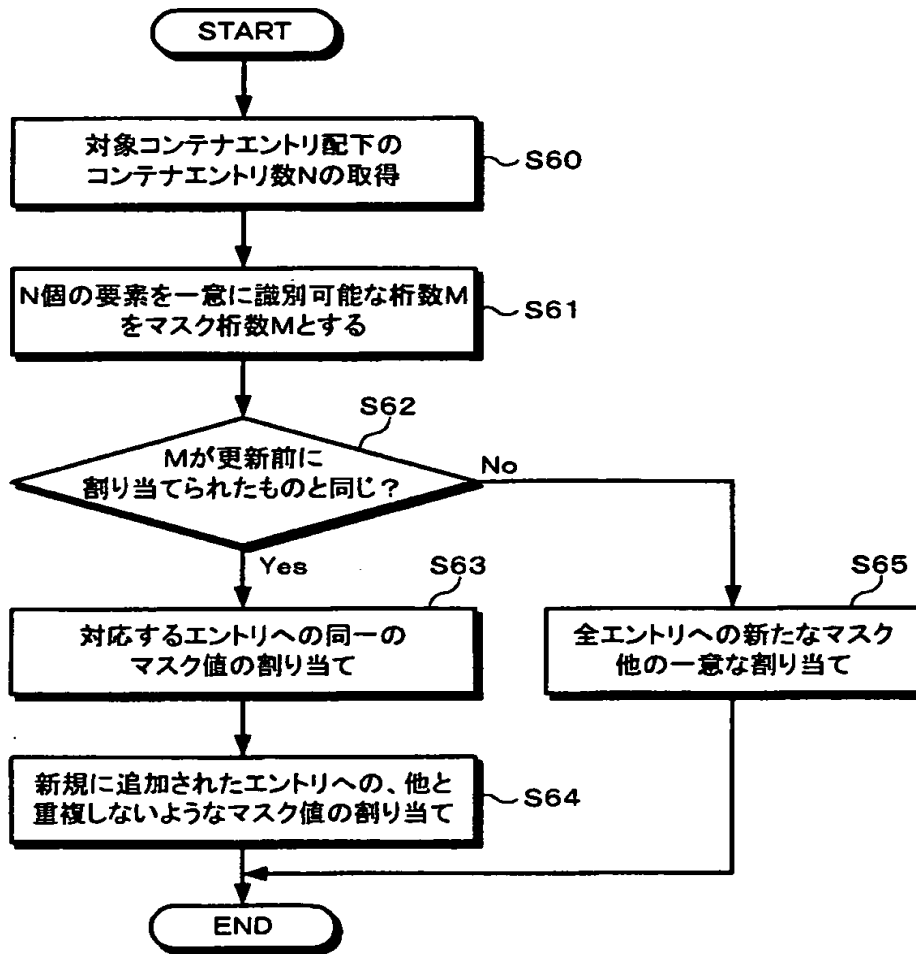
【図16】



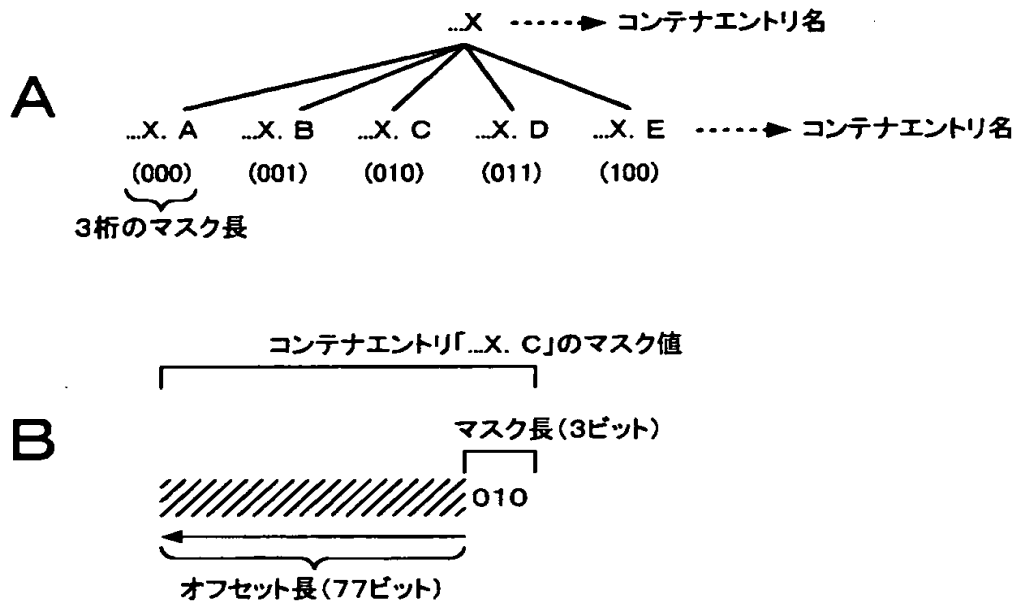
【図 1 7】



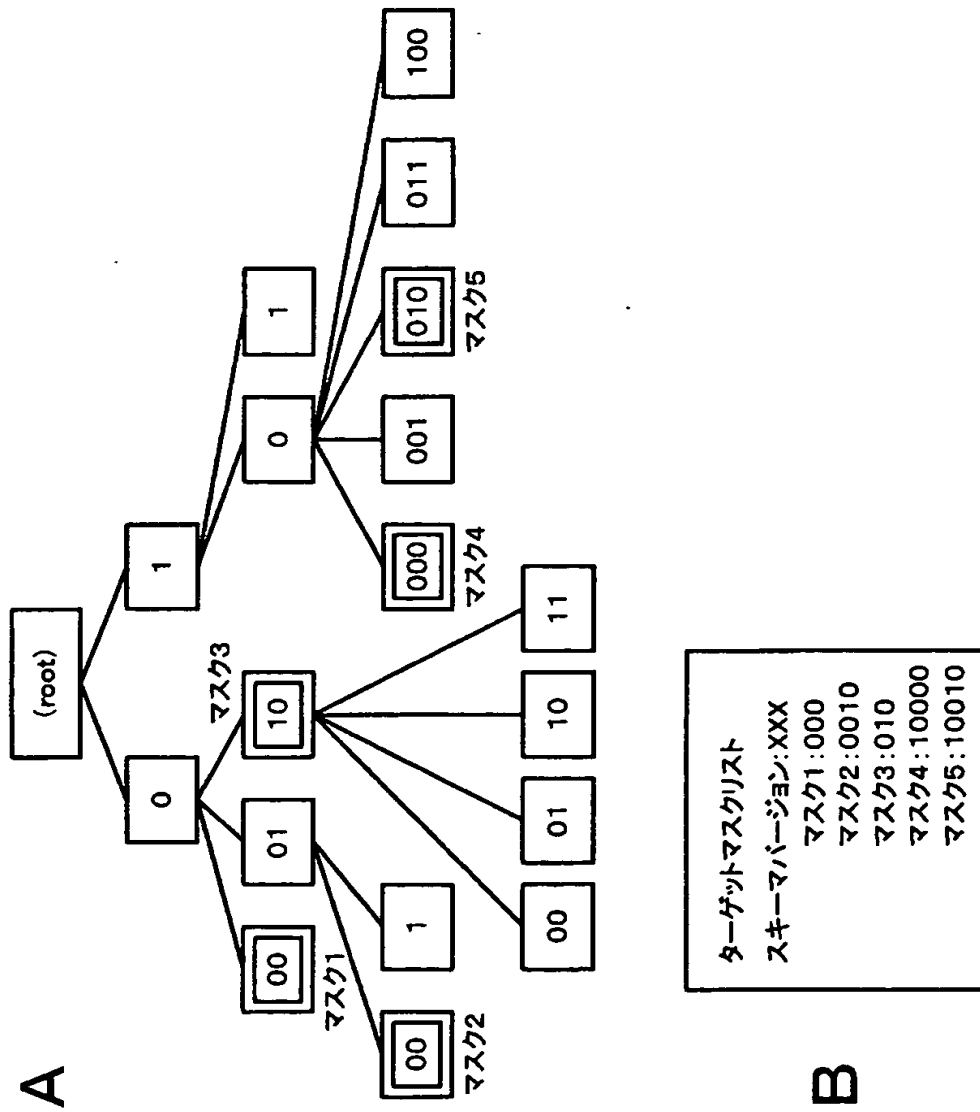
【図18】



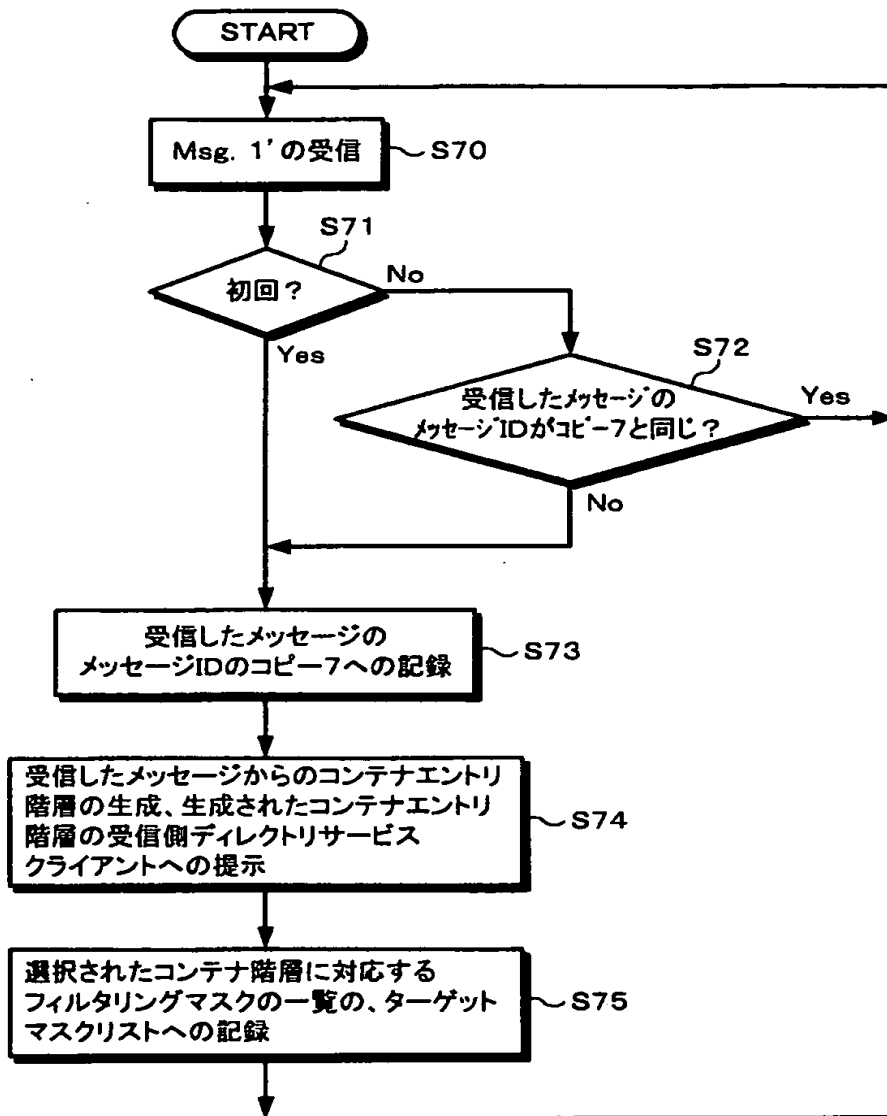
【図 1 9】



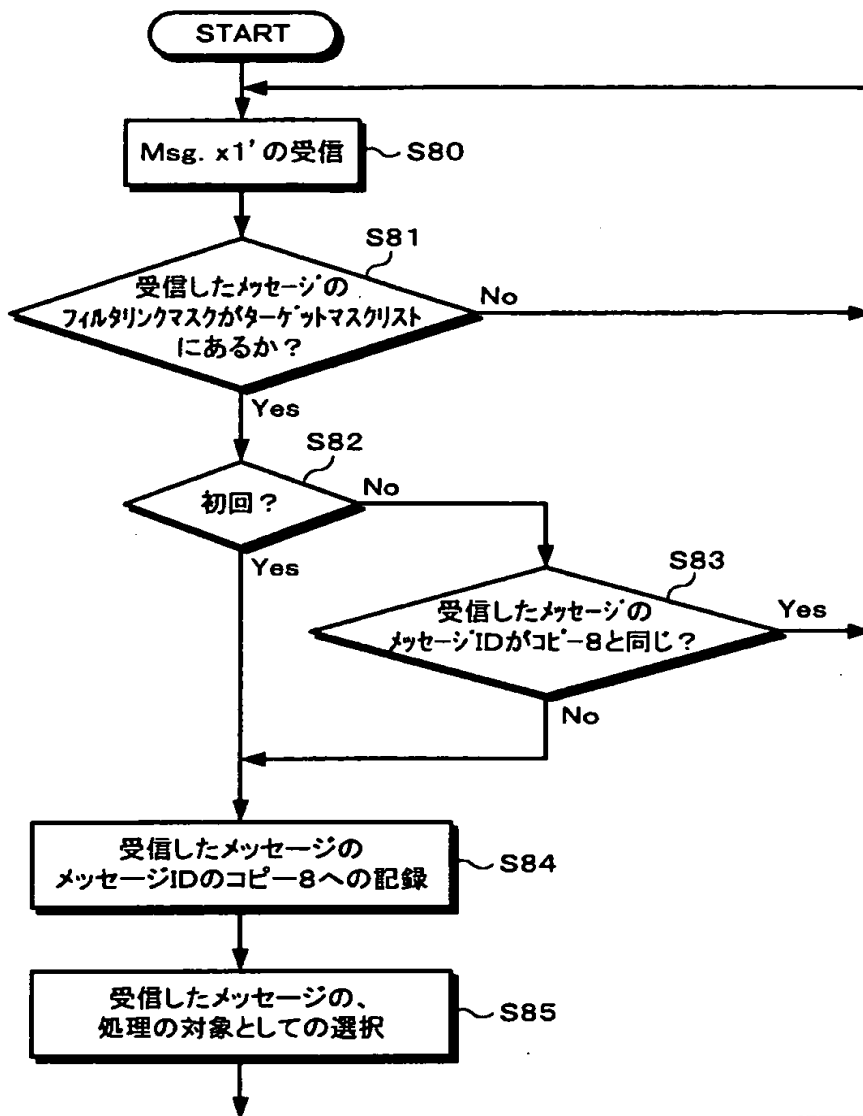
【図 2 0】



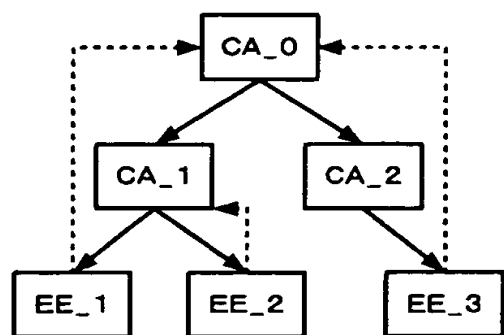
【図 2 1】



【図 2 2】



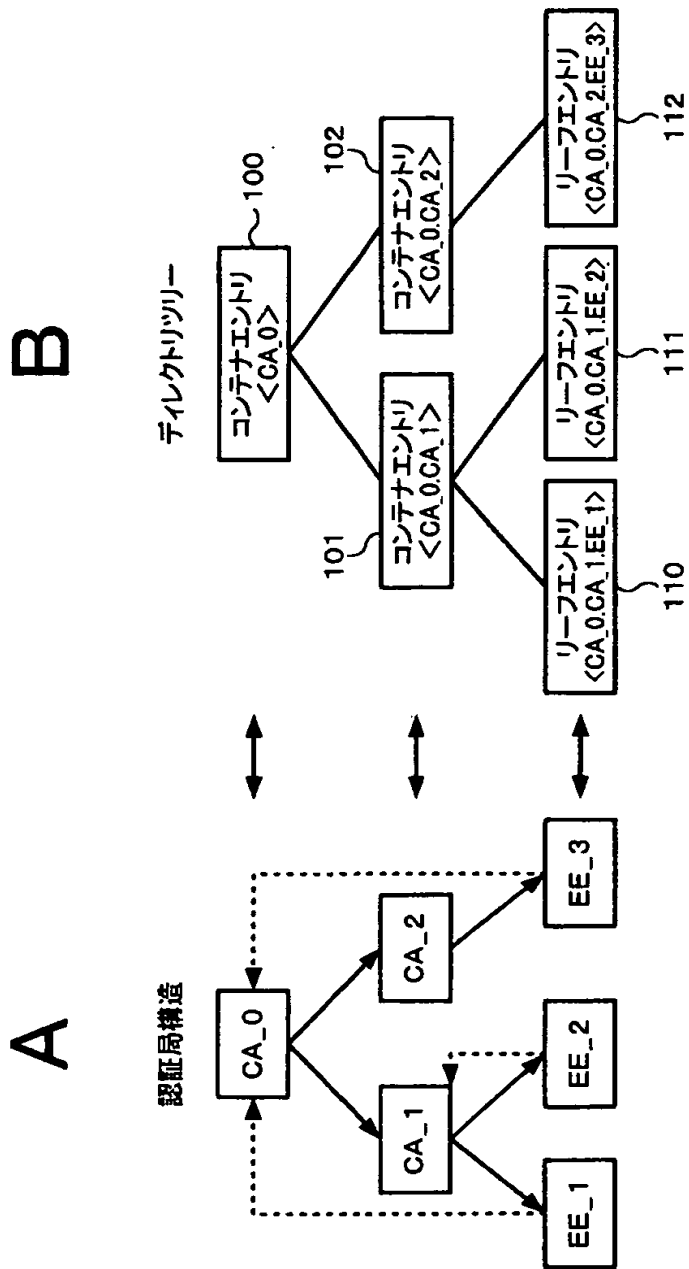
【図 2 3】



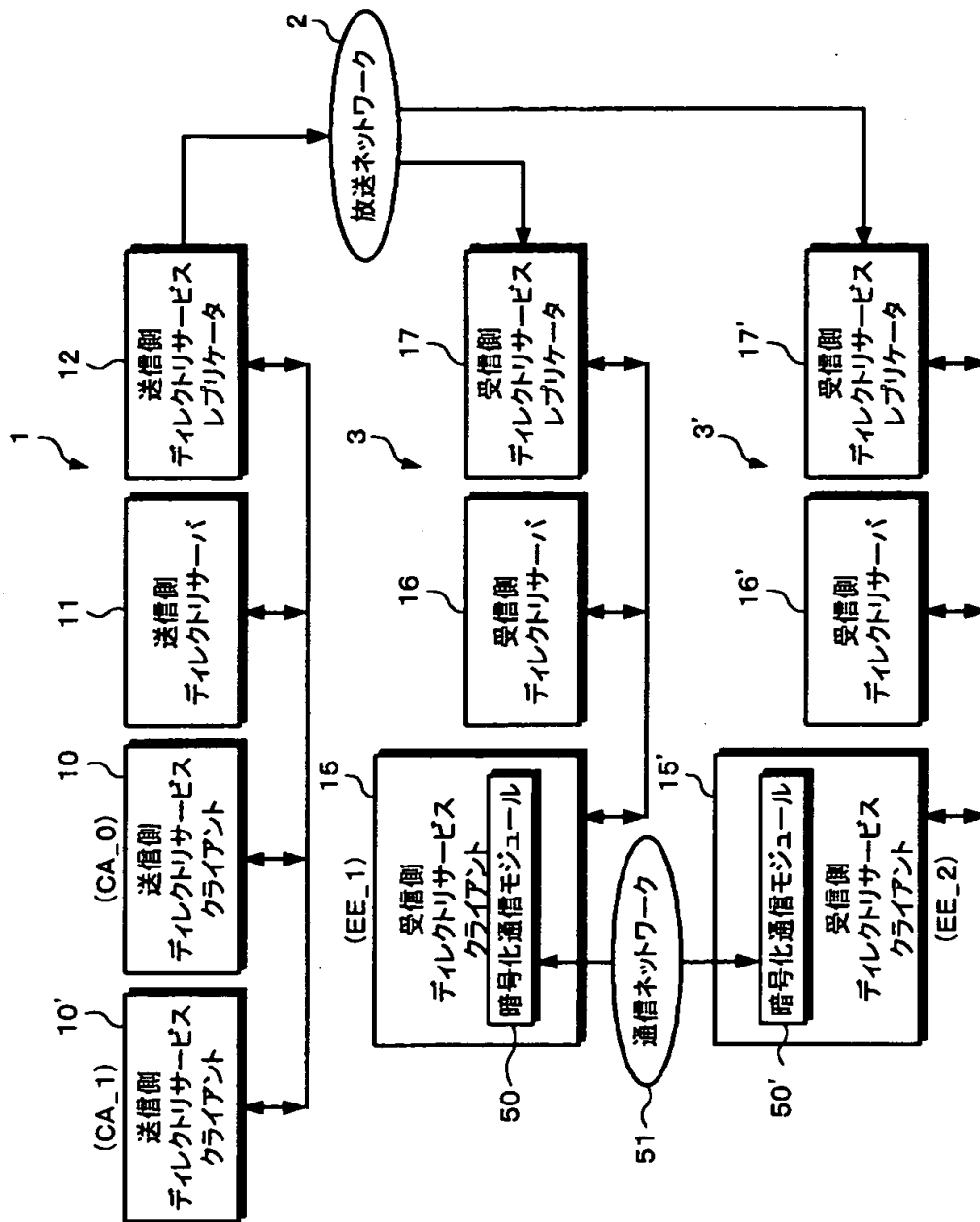
ルート認証局として信頼する : - - - - ->

公開鍵証明書を発行する : ———>

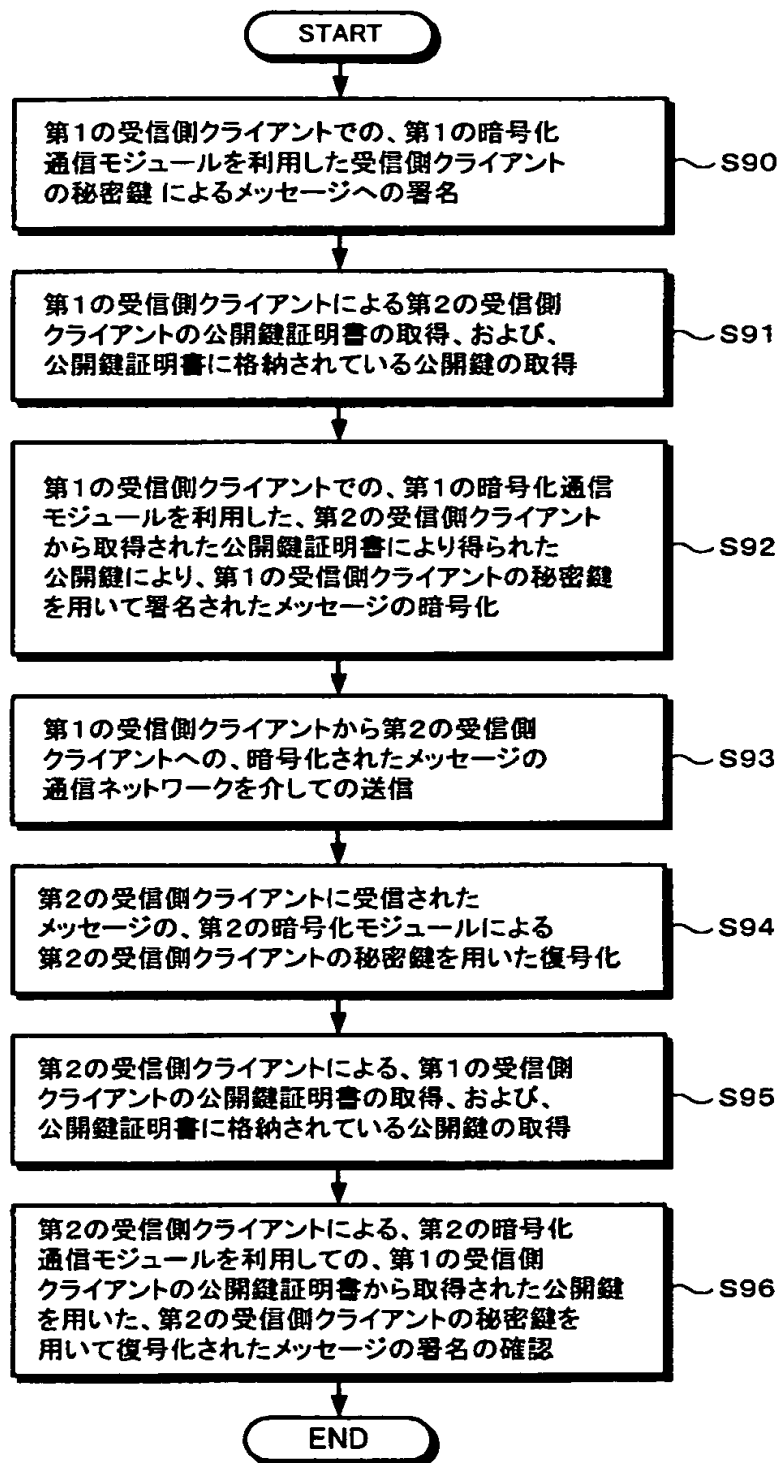
【図 2 4】



【図 25】

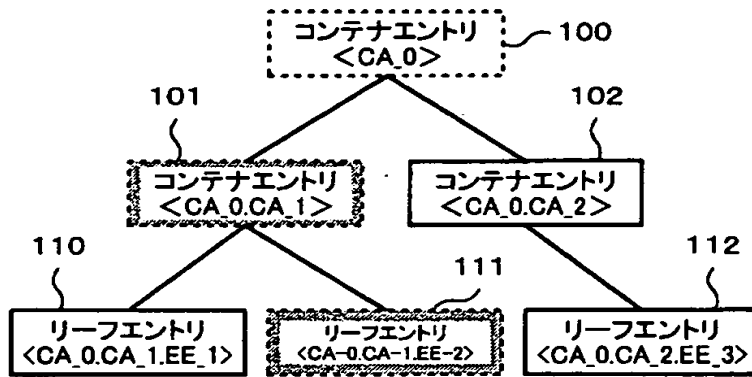


【図 2 6】

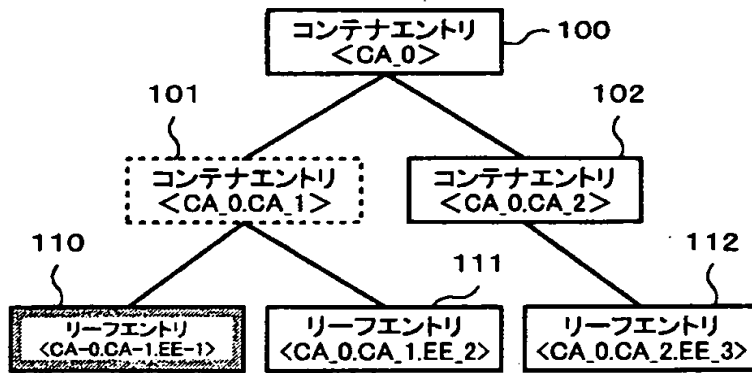


【図 27】

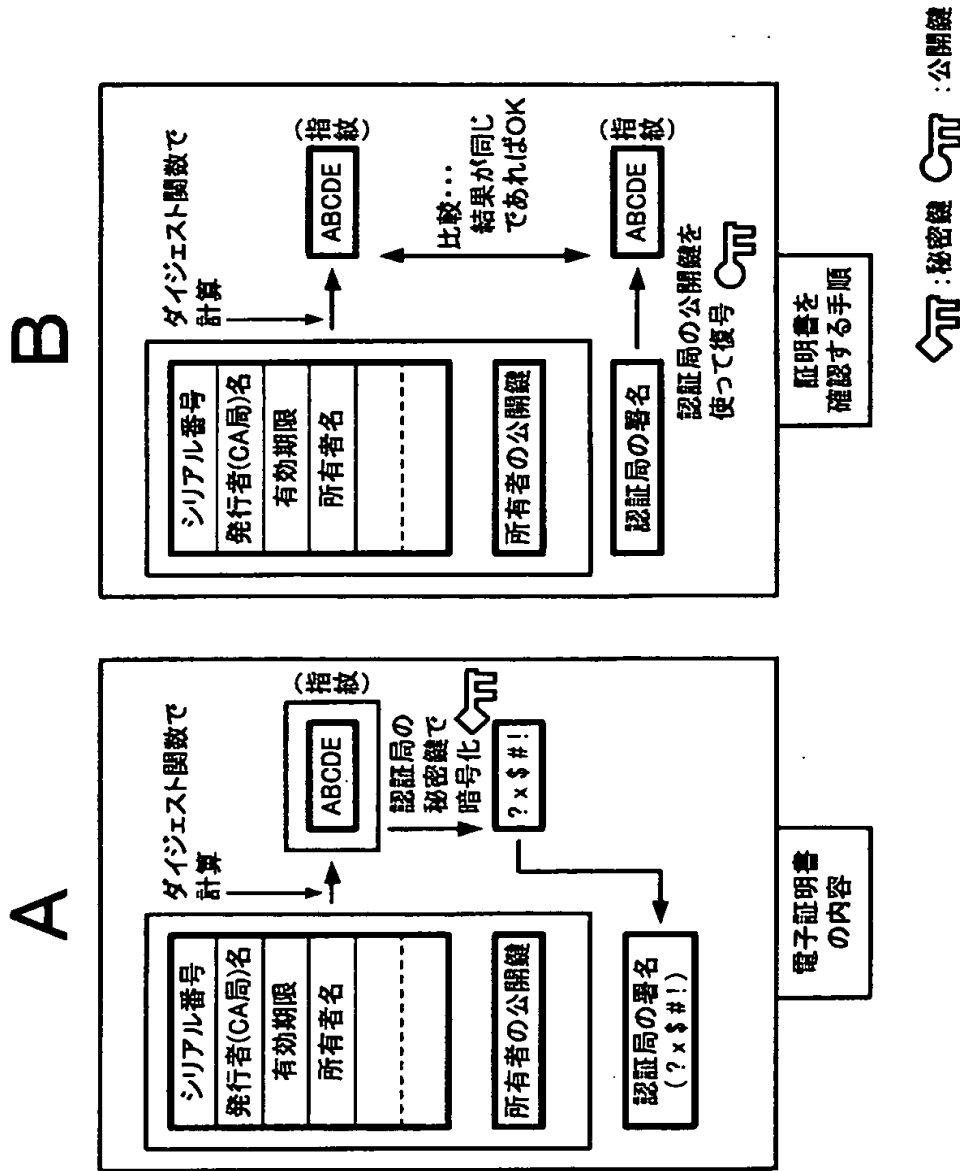
A



B



【図 2 8】



【書類名】 要約書

【要約】

【課題】 P K I ディレクトリサーバから P K I ディレクトリクライアントへ公開鍵証明書失効情報の通知を、効率よく行うことができるようにする。

【解決手段】 認証局構造における認証局情報をコンテナエントリに、エンドエンティティ情報をリーフエントリに対応付けて、認証局構造をディレクトリツリーに当てはめる。各エントリの属性として、最新の公開鍵証明書そのもの或いはその取得方法、ならびに、シリアル番号とを持たせる。ある証明書が失効し新しく証明書が発行されたら、新たに発行された証明書とそのシリアル番号とをエントリに格納する。所定の時間経過後に、当該証明書がある U R L に置くと共に、エントリに格納された証明書を U R L 情報と置き換える。一方、受信側では、必要な証明書を取得するための証明書パスに基づき、フィルタリングマスクを設定する。送信側から送信されたディレクトリ構造は、このフィルタリングマスクにより選択的に受信され、受信側のディレクトリを変更する。送信側から U R L 情報とシリアル番号が格納されたディレクトリツリーが繰り返し送信され、受信側では、フィルタリングマスクで選択されたエントリだけを更新する。最新の公開鍵証明書の参照が分散される。

【選択図】 図 2 4

出 願 人 履 歴 情 報

識別番号 [000002185]

1. 変更年月日 1990年 8月30日
[変更理由] 新規登録
住 所 東京都品川区北品川6丁目7番35号
氏 名 ソニー株式会社